OpenShift Container Platform 4.6

安装

安装并配置 OpenShift Container Platform 集群

Last Updated: 2023-02-27

OpenShift Container Platform 4.6 安装

安装并配置 OpenShift Container Platform 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

Copyright © 2022 | You need to change the HOLDER entity in the en-US/Installing.ent file |.

The text of and illustrations in this document are licensed by Red Hat under a Creative Commons Attribution–Share Alike 3.0 Unported license ("CC-BY-SA"). An explanation of CC-BY-SA is available at
http://creativecommons.org/licenses/by-sa/3.0/
. In accordance with CC-BY-SA, if you distribute this document or an adaptation of it, you must provide the URL for the original version.

Red Hat, as the licensor of this document, waives the right to enforce, and agrees not to assert, Section 4d of CC-BY-SA to the fullest extent permitted by applicable law.

Red Hat, Red Hat Enterprise Linux, the Shadowman logo, the Red Hat logo, JBoss, OpenShift, Fedora, the Infinity logo, and RHCE are trademarks of Red Hat, Inc., registered in the United States and other countries.

Linux ® is the registered trademark of Linus Torvalds in the United States and other countries.

Java ® is a registered trademark of Oracle and/or its affiliates.

XFS ® is a trademark of Silicon Graphics International Corp. or its subsidiaries in the United States and/or other countries.

MySQL ® is a registered trademark of MySQL AB in the United States, the European Union and other countries.

Node.js ® is an official trademark of Joyent. Red Hat is not formally related to or endorsed by the official Joyent Node.js open source or commercial project.

The OpenStack ® Word Mark and OpenStack logo are either registered trademarks/service marks or trademarks/service marks of the OpenStack Foundation, in the United States and other countries and are used with the OpenStack Foundation's permission. We are not affiliated with, endorsed or sponsored by the OpenStack Foundation, or the OpenStack community.

All other trademarks are the property of their respective owners.

摘要

本文档提供有关安装和配置 OpenShift Container Platform 的信息。

目录

第 1 章 为断开连接的安装 MIRROR 镜像

您可以使用本节中的步骤确保集群只使用满足您机构对外部内容控制的容器镜像。在受限网络中置备的基础架构上安装集群前，您必须将所需的容器镜像镜像(mirror)到那个环境中。要镜像容器镜像，您必须有一个 registry 才能进行镜像(mirror)。

重要

您必须可以访问互联网来获取所需的容器镜像。在这一流程中，您要将镜像 registry 放在可访问您的网络以及互联网的镜像(mirror)主机上。如果您没有镜像主机的访问权限，请使用 镜像 Operator 目录 流程将镜像复制到可跨网络界限的设备中。

1.1. 先决条件

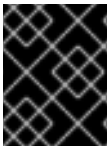
- 您必须在托管 OpenShift Container Platform 集群的位置（如以下 registry 之一）中有一个支持 Docker v2-2 的容器镜像 registry：
  - Red Hat Quay
  - JFrog Artifactory
  - Sonatype Nexus 仓库
  - Harbor

  如果您有 Red Hat Quay 权利，请参阅有关部署 Red Hat Quay 以了解概念验证的文档，或使用 Quay Operator。如果您需要额外的帮助来选择并安装 registry，请联络您的销售代表或红帽支持。
- 如果您还没有容器镜像 registry 的现有解决方案，则会为 Red Hat OpenShift 提供一个 OpenShift Container Platform 订阅(mirror) registry。*Red Hat OpenShift 的镜像 registry* 包含在您的订阅中，它是一个小型容器 registry，可用于在断开连接的安装中镜像 OpenShift Container Platform 所需的容器镜像。

1.2. 关于镜像 REGISTRY

您可以镜像 OpenShift Container Platform 安装所需的镜像，以及容器镜像 registry 的后续产品更新，如 Red Hat Quay、JFrog Artifactory、Sonatype Nexus Repository 或 Harbor。如果您无法访问大型容器 registry，可以使用 *Red Hat OpenShift 的镜像 registry（OpenShift Container Platform 订阅中包含的小型容器 registry）*。

您可以使用支持 Docker v2-2 *的任何容器 registry，如 Red Hat Quay、Red Hat OpenShift 的镜像 registry*、Artifactory、Sonatype Nexus Repository 或 Harbor。无论您所选 registry 是什么，都会将互联网上红帽托管站点的内容镜像到隔离的镜像 registry 相同。镜像内容后，您要将每个集群配置为从镜像 registry 中检索此内容。



重要

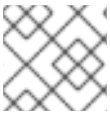
OpenShift Container Platform 集群的内部 registry 不能用作目标 registry，因为它不支持没有标签的推送（在镜像过程中需要这个功能）。

如果选择不是 *Red Hat OpenShift 的镜像 registry* 的容器 registry，则您必须可以被您置备的集群中的每台机器访问。如果 registry 无法访问，安装、更新或常规操作（如工作负载重新定位）可能会失败。因此，您必须以高度可用的方式运行镜像 registry，镜像 registry 至少必须与 OpenShift Container Platform

集群的生产环境可用性相匹配。

使用 OpenShift Container Platform 镜像填充镜像 registry 时，可以遵循以下两种情况。如果您的主机可以同时访问互联网和您的镜像 registry，而不能访问您的集群节点，您可以直接从该机器中镜像该内容。这个过程被称为 *连接的镜像(mirror)*。如果没有这样的主机，则必须将该镜像文件镜像到文件系统中，然后将该主机或者可移动介质放入受限环境中。这个过程被称为 *断开连接的镜像*。

对于已镜像的 registry，若要查看拉取镜像的来源，您必须查看 **Trying 以访问** CRI-O 日志中的日志条目。查看镜像拉取源的其他方法（如在节点上使用 **crictl images** 命令）显示非镜像镜像名称，即使镜像是从镜像位置拉取的。



注意

红帽不支持使用 OpenShift Container Platform 测试第三方 registry。

其他信息

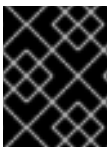
有关查看 CRI-O 日志以查看镜像源的详情，请参阅查看镜像拉取源。

1.3. 准备您的镜像主机

执行镜像步骤前，必须准备主机以检索内容并将其推送到远程位置。

1.3.1. 通过下载二进制文件安装 OpenShift CLI

您需要安装 CLI（**oc**） 来使用命令行界面与 OpenShift Container Platform 进行交互。您可在 Linux、Windows 或 macOS 上安装 **oc**。



重要

如果安装了旧版本的 **oc**，则无法使用 OpenShift Container Platform 4.6 中的所有命令。下载并安装新版本的 **oc**。

1.3.1.1. 在 Linux 上安装 OpenShift CLI

您可以按照以下流程在 Linux 上安装 OpenShift CLI（**oc**）二进制文件。

流程

1. 进入到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 在 **Version** 下拉菜单中选择相应的版本。
3. 单击 **OpenShift v4.6 Linux 客户端** 条目旁边的 **Download Now**，再保存文件。
4. 解包存档：

```
$ tar xvzf <file>
```

5. 把 **oc** 二进制代码放到 **PATH** 中的目录中。
   执行以下命令可以查看当前的 **PATH** 设置：

```
$ echo $PATH
```

安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
$ oc <command>
```

1.3.1.2. 在 Windows 上安装 OpenShift CLI

您可以按照以下流程在 Windows 上安装 OpenShift CLI（**oc**）二进制代码。

流程

1. 进入到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 在 **Version** 下拉菜单中选择相应的版本。
3. 单击 **OpenShift v4.6 Windows 客户端**条目旁边的 **Download Now**，再保存文件。
4. 使用 ZIP 程序解压存档。
5. 把 **oc** 二进制代码放到 **PATH** 中的目录中。
   要查看您的 **PATH**，请打开命令提示窗口并执行以下命令：

   ```
   C:\> path
   ```

安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
C:\> oc <command>
```

1.3.1.3. 在 macOS 上安装 OpenShift CLI

您可以按照以下流程在 macOS 上安装 OpenShift CLI（**oc**）二进制代码。

流程

1. 进入到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 在 **Version** 下拉菜单中选择相应的版本。
3. 单击 **OpenShift v4.6 MacOSX 客户端**条目旁边的 **Download Now**，再保存文件。
4. 解包和解压存档。
5. 将 **oc** 二进制文件移到 PATH 的目录中。
   要查看您的 **PATH**，打开一个终端窗口并执行以下命令：

   ```
   $ echo $PATH
   ```

安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
$ oc <command>
```

1.4. 配置允许对容器镜像进行镜像的凭证

创建容器镜像 registry 凭证文件，允许将红帽的镜像镜像到您的镜像环境中。

警告

安装集群时不要使用此镜像 registry 凭据文件作为 pull secret。如果在安装集群时提供此文件，集群中的所有机器都将具有镜像 registry 的写入权限。

警告

此过程需要您可以对镜像 registry 上的容器镜像 registry 进行写操作，并将凭证添加到 registry pull secret。

先决条件

- 配置了一个镜像（mirror） registry 在受限网络中使用。
- 您在镜像 registry 中标识了镜像仓库的位置，以将容器镜像镜像(mirror)到这个位置。
- 您置备了一个镜像 registry 帐户，允许将镜像上传到该镜像仓库。

流程

在安装主机上完成以下步骤：

1. 从 Red Hat OpenShift Cluster Manager 下载 **registry.redhat.io** 的 pull secret，并将它保存到 **.json** 文件中。
2. 为您的镜像 registry 生成 base64 编码的用户名和密码或令牌：

```
$ echo -n '<user_name>:<password>' | base64 -w0 (1)
BGVtbYk3ZHAtqXs=
```

(1) 通过 **<user_name>** 和 **<password>** 指定 registry 的用户名和密码。

3. 以 JSON 格式创建您的 pull secret 副本：

```
$ cat ./pull-secret.text | jq .  > <path>/<pull_secret_file_in_json> (1)
```

(1) 指定到存储 pull secret 的文件夹的路径，以及您创建的 JSON 文件的名称。

该文件类似于以下示例：

```
{
  "auths": {
```

```
    "cloud.openshift.com": {
      "auth": "b3BlbnNo...",
      "email": "you@example.com"
    },
    "quay.io": {
      "auth": "b3BlbnNo...",
      "email": "you@example.com"
    },
    "registry.connect.redhat.com": {
      "auth": "NTE3Njg5Nj...",
      "email": "you@example.com"
    },
    "registry.redhat.io": {
      "auth": "NTE3Njg5Nj...",
      "email": "you@example.com"
    }
  }
}
```

4. 编辑新文件并添加描述 registry 的部分：

```
"auths": {
  "<mirror_registry>": { 1
    "auth": "<credentials>", 2
    "email": "you@example.com"
  },
```

1 对于 **<mirror_registry>**，指定 registry 域名，以及您的镜像 registry 用来提供内容的可选端口。例如： **registry.example.com** 或 **registry.example.com:8443**

2 使用 **<credentials>** 为您的镜像 registry 指定 base64 编码的用户名和密码。

该文件类似于以下示例：

```
{
  "auths": {
    "registry.example.com": {
      "auth": "BGVtbYk3ZHAtqXs=",
      "email": "you@example.com"
    },
    "cloud.openshift.com": {
      "auth": "b3BlbnNo...",
      "email": "you@example.com"
    },
    "quay.io": {
      "auth": "b3BlbnNo...",
      "email": "you@example.com"
    },
    "registry.connect.redhat.com": {
      "auth": "NTE3Njg5Nj...",
      "email": "you@example.com"
    },
    "registry.redhat.io": {
      "auth": "NTE3Njg5Nj...",
```

```
      "email": "you@example.com"
    }
  }
}
```

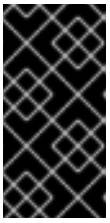
1.5. RED HAT OPENSHIFT 的镜像(MIRROR)REGISTRY

Red Hat OpenShift 的镜像 registry 是一个小且简化的容器 registry，作为目标，用于为断开连接的安装镜像(mirror)的 OpenShift Container Platform 所需的容器镜像。

如果您已有容器镜像 registry（如 Red Hat Quay），您可以跳过这些步骤，并直接 镜像 OpenShift Container Platform 镜像存储库。

先决条件

- OpenShift Container Platform 订阅。
- 安装了 Podman 3.3 和 OpenSSL 的 Red Hat Enterprise Linux(RHEL)8。
- Red Hat Quay 服务的完全限定域名，它必须通过 DNS 服务器解析。
- 目标主机上的免密码 **sudo** 访问。
- 目标主机上的基于密钥的 SSH 连接。为本地安装自动生成 SSH 密钥。对于远程主机，您必须生成自己的 SSH 密钥。
- 2 个或更多 vCPU。
- 8 GB RAM。
- 关于 OpenShift Container Platform 4.6 发行镜像的 6.8 GB，或大约 696 GB OpenShift Container Platform 4.6 发行镜像和 OpenShift Container Platform 4.6 Red Hat Operator 镜像。建议每个流或更长时间最多 1 TB。



重要

这些要求基于本地测试结果，且只测试了发行镜像和 Operator 镜像。存储要求可能会因您的组织的需求而有所不同。有些用户可能需要更多空间，例如当它们镜像多个 z-streams 时。您可以使用标准 Red Hat Quay 功能删除不必要的镜像并腾出空间。

1.5.1. Red Hat OpenShift 简介的镜像(mirror)registry

对于断开连接的 OpenShift Container Platform 部署，需要一个容器 registry 来安装集群。要在这样的集群中运行 production-grade registry 服务，您必须创建一个单独的 registry 部署来安装第一个集群。此 *需要 Red Hat OpenShift 的镜像 registry*，并包括在每个 OpenShift 订阅中。它可用于在 OpenShift 控制台 **Downloads** 页面下载。

Red Hat OpenShift 的镜像 registry 允许用户使用 ***mirror*** *-registry* 命令行界面(CLI)工具安装一个较小的 Red Hat Quay 版本及其所需的组件。*Red Hat OpenShift 的镜像 registry* 会自动部署预先配置的本地存储和本地数据库。它还包括自动生成的用户凭证和访问权限，其中只有一个输入集，且不会启动的额外配置选项。

Red Hat OpenShift 的镜像 registry 提供了一个预先确定的网络配置，并在成功时报告部署的组件凭证并访问 URL。另外还提供了一组有限的可选配置输入，如完全限定域名(FQDN)服务、超级用户名称和密

码，以及自定义 TLS 证书。这为用户提供了一个容器 registry，以便在受限网络环境中运行 OpenShift Container Platform 时，轻松创建所有 OpenShift Container Platform 发行版本内容的离线镜像。

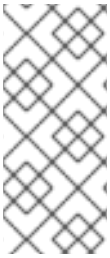
Red Hat OpenShift 的镜像 registry 仅限于托管安装断开连接的 OpenShift Container Platform 集群（如发行镜像或 Red Hat Operator 镜像）所需的镜像。它使用 Red Hat Enterprise Linux(RHEL)机器上的本地存储，而 RHEL 支持的存储由 *Red Hat OpenShift 的镜像 registry 支持*。由客户创建的内容不应由 *Red Hat OpenShift 的镜像 registry* 托管。

与 Red Hat Quay 不同，*Red Hat OpenShift 的镜像 registry* 不是高度可用的 registry，且只支持本地文件系统存储。不建议在多个集群中使用 *Red Hat OpenShift 镜像 registry*，因为有多个集群可以在更新集群时创建单点故障。建议利用 *Red Hat OpenShift 的镜像 registry* 安装可托管生产环境、高可用性 registry（如 Red Hat Quay）的集群，以便将 OpenShift Container Platform 内容提供给其他集群。

如果在安装环境中已有另一个容器 *registry，则为 Red Hat OpenShift* 使用镜像 registry 是可选的。

1.5.2. 使用 Red Hat OpenShift 的镜像 registry 在本地主机上镜像(mirror)

此流程解释了如何使用 ***mirror-registry* 安装程序工具在本地主机上为 Red Hat OpenShift 安装镜像 registry**。这样，用户可以创建在端口 443 上运行的本地主机 registry，以存储 OpenShift Container Platform 镜像的镜像。



注意

使用 ***mirror-registry* CLI 工具安装 Red Hat OpenShift 的镜像 registry** 对 机器进行了几个更改。安装后，会创建一个 **/etc/quay-install** 目录，该目录具有安装文件、本地存储和配置捆绑包。如果部署目标是本地主机，则生成可信 SSH 密钥，并且设置主机计算机上的 systemd 文件，以确保容器运行时持久。另外，会创建一个名为 **init** 的初始用户，并自动生成的密码。所有访问凭证都会在安装例程的末尾打印。

流程

1. 为 OpenShift 控制台 **Downloads** 页面找到的 *Red Hat OpenShift 的最新版本下载* ***mirror***-*registry.tar.gz* 软件包。

2. 使用 ***mirror-registry* 工具，在本地主机上安装 Red Hat OpenShift 的镜像 registry**。有关可用标志的完整列表，请参阅 "mirror registry for Red Hat OpenShift flags"。

```
$ sudo ./mirror-registry install \
  --quayHostname <host_example_com> \
  --quayRoot <example_directory_name>
```

3. 运行以下命令，使用安装期间生成的用户名和密码登录 registry：

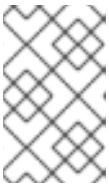
```
$ podman login --authfile pull-secret.txt \
  -u init \
  -p <password> \
  <host_example_com>:8443> \
  --tls-verify=false 1
```

1 您可以通过将您的系统配置为信任生成的 rootCA 证书来避免运行 **--tls-verify=false**。如需更多信息，请参阅"使用 SSL 保护到 Red Hat Quay 的连接"和"配置系统以信任证书认证机构"。

注意

您还可以在安装后通过 **https://<host.example.com>:8443** 访问 UI 登录。

4. 您可以在登录后镜像 OpenShift Container Platform 镜像。根据您的需要，请参阅本文档的"镜像 OpenShift Container Platform 镜像存储库"或"镜像 Operator 目录"部分。



注意

如果因为存储层问题导致 *Red Hat OpenShift 镜像存储了镜像 registry* 存在问题，您可以在更稳定的存储上对 OpenShift Container Platform 镜像重新镜像(mirror)或重新安装 registry。

1.5.3. 使用 Red Hat OpenShift 的镜像 registry 在远程主机上镜像(mirror)

此流程解释了如何使用 ***mirror-registry*** **工具在远程主机上为 Red Hat OpenShift 安装镜像 registry**。这样，用户可以创建 registry 来保存 OpenShift Container Platform 镜像的镜像。

注意

使用 ***mirror-registry*** **CLI 工具安装 Red Hat OpenShift 的镜像 registry** 对 机器进行了几个更改。安装后，会创建一个 **/etc/quay-install** 目录，该目录具有安装文件、本地存储和配置捆绑包。如果部署目标是本地主机，则生成可信 SSH 密钥，并且设置主机计算机上的 systemd 文件，以确保容器运行时持久。另外，会创建一个名为 **init** 的初始用户，并自动生成的密码。所有访问凭证都会在安装例程的末尾打印。

流程

1. 为 OpenShift 控制台 **Downloads** 页面找到的 *Red Hat OpenShift 的最新版本下载* ***mirror***-*registry.tar.gz* 软件包。

2. 使用 ***mirror-registry*** **工具，在本地主机上安装 Red Hat OpenShift 的镜像 registry**。有关可用标志的完整列表，请参阅 "mirror registry for Red Hat OpenShift flags"。

```
$ sudo ./mirror-registry install -v \
  --targetHostname <host_example_com> \
  --targetUsername <example_user> \
  -k ~/.ssh/my_ssh_key \
  --quayHostname <host_example_com> \
  --quayRoot <example_directory_name>
```

3. 运行以下命令，使用安装期间生成的用户名和密码登录镜像 registry：

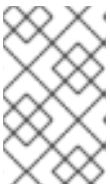
```
$ podman login --authfile pull-secret.txt \
  -u init \
  -p <password> \
  <host_example_com>:8443> \
  --tls-verify=false 1
```

1 您可以通过将您的系统配置为信任生成的 rootCA 证书来避免运行 **--tls-verify=false**。如需更多信息，请参阅"使用 SSL 保护到 Red Hat Quay 的连接"和"配置系统以信任证书认证机构"。

注意

您还可以在安装后通过 **https://<host.example.com>:8443** 访问 UI 登录。

4. 您可以在登录后镜像 OpenShift Container Platform 镜像。根据您的需要，请参阅本文档的"镜像 OpenShift Container Platform 镜像存储库"或"镜像 Operator 目录"部分。



注意

如果因为存储层问题导致 *Red Hat OpenShift 镜像存储了镜像 registry* 存在问题，您可以在更稳定的存储上对 OpenShift Container Platform 镜像重新镜像(mirror)或重新安装 registry。

1.6. 为 RED HAT OPENSHIFT 升级镜像 REGISTRY

- 您可以运行以下命令来 *从本地主机升级 Red Hat OpenShift 的镜像 registry* ：

```
$ sudo ./mirror-registry upgrade
```

注意

- 使用 **./ *mirror-registry 升级* Red Hat OpenShift 的镜像 registry** 用户必须与创建镜像 registry 时使用的凭证相同。例如，如果使用 **--quayHostname *<host_example_com&* gt; 和 --quayRoot <example_directory_name> 安装了 Red Hat OpenShift** 的镜像 registry，则必须包括该字符串才能正确地升级镜像 registry。

1.6.1. 为 Red Hat OpenShift 卸载镜像 registry

- 您可以运行以下命令来从本地主机 *中卸载 Red Hat OpenShift 的镜像 registry* ：

```
$ sudo ./mirror-registry uninstall -v \
  --quayRoot <example_directory_name>
```

注意

- 删除 *Red Hat OpenShift 的镜像 registry* 会在删除前提示用户。您可以使用 **--autoApprove** 来跳过此提示。
- 使用 **--quayRoot** 标志 *为 Red Hat OpenShift 安装镜像 registry* 的用户必须在卸载时包括 **--quayRoot** 标志。例如，如果使用 **--quayRoot example_directory_name** 安装了 *Red Hat OpenShift 的镜像 registry*，则必须包括该字符串才能正确地卸载镜像 registry。

1.6.2. Red Hat OpenShift 标记的镜像(mirror)registry

以下标记可用于 *Red Hat OpenShift 的镜像 registry* ：

标记	描述

标记	描述
--autoApprove	禁用交互式提示的布尔值。如果设置为 **true**，则在卸载镜像 registry 时自动删除 **quayRoot** 目录。如果未指定，则默认为 **false**。
--initPassword	在 Quay 安装过程中创建的 init 用户的密码。必须至少包含八个字符，且不包含空格。
--initUser 字符串	显示初始用户的用户名。如果未指定，则默认为 **init**。
--quayHostname	客户端用来联系 registry 的镜像 registry 的完全限定域名。等同于 Quay **config.yaml** 中的 **SERVER_HOSTNAME**。必须由 DNS 解析。如果未指定，则默认为 **<targetHostname>:8443**。[1]
--quayRoot, -r	保存容器镜像层和配置数据的目录，包括 **rootCA.key**、rootCA**.pem** 和 **rootCA.srl** 证书。OpenShift Container Platform 4.6 发行镜像需要 6.8 GB，或大约 696 GB OpenShift Container Platform 4.6 发行镜像和 OpenShift Container Platform 4.6 Red Hat Operator 镜像。如果未指定，则默认为 **/etc/quay-install**。
--ssh-key,-k	SSH 身份密钥的路径。如果未指定，则默认为 **~/.ssh/quay_installer**。
--sslCert	SSL/TLS 公钥/证书的路径。默认为 **{quayRoot}/quay-config**，并在未指定时自动生成。
--sslCheckSkip	跳过对 **config.yaml** 文件中的 **SERVER_HOSTNAME** 的检查证书主机名。[2]
--sslKey	用于 HTTPS 通信的 SSL/TLS 私钥路径。默认为 **{quayRoot}/quay-config**，并在未指定时自动生成。
--targetHostname,-H	要安装 Quay 的目标的主机名。默认为 **$HOST**，如本地主机（如果未指定）。
--targetUsername,-u	目标主机上的用户，将用于 SSH。默认为 **$USER**，例如，如果未指定，则默认为当前用户。
--verbose,-v	显示调试日志和 Ansible playbook 输出。
--version	显示 *Red Hat OpenShift 的镜像 registry* 的版本。

1. 如果您的系统的公共 DNS 名称与本地主机名不同，则必须修改 **--quayHostname**。
2. 当镜像 registry 在代理后面设置时，会使用 **--sslCheckSkip**，并且公开的主机名与内部 Quay 主机名不同。当用户不希望在安装过程中对提供的 Quay 主机名验证证书时，也可以使用它。

其他资源

- 使用 SSL 保护到 Red Hat Quay 的连接

- 将系统配置为信任证书颁发机构
- 镜像 OpenShift Container Platform 镜像存储库
- 对 Operator 目录进行镜像(mirror)

1.7. 镜像 OPENSHIFT CONTAINER PLATFORM 镜像存储库

镜像要在集群安装或升级过程中使用的 OpenShift Container Platform 镜像仓库。

先决条件

- 您的镜像主机可访问互联网。
- 您已将镜像 registry 配置为在受限网络中使用，并可访问您配置的证书和凭证。
- 您已从 Red Hat OpenShift Cluster Manager 下载了 pull secret，并已修改为包含镜像存储库身份验证信息。
- 如果您使用没有设置 Subject Alternative Name 的自签名证书，则必须在这个过程中使用 **GODEBUG=x509ignoreCN=0** 前执行 **oc** 命令。如果没有设置此变量，**oc** 命令会失败并显示以下错误：

```
x509: certificate relies on legacy Common Name field, use SANs or temporarily enable
Common Name matching with GODEBUG=x509ignoreCN=0
```

流程

在镜像主机上完成以下步骤：

1. 查看 OpenShift Container Platform 下载页面，以确定您要安装的 OpenShift Container Platform 版本，并决定 Repository Tags 页中的相应标签（tag）。
2. 设置所需的环境变量：
   a. 导出发行版本信息：

```
$ OCP_RELEASE=<release_version>
```

   对于 **<release_version>**，请指定与 OpenShift Container Platform 版本对应的标签，用于您的架构，如 **4.5.4**。

   b. 导出本地 registry 名称和主机端口：

```
$ LOCAL_REGISTRY='<local_registry_host_name>:<local_registry_host_port>'
```

   对于 **<local_registry_host_name>**，请指定镜像存储库的 registry 域名；对于 **<local_registry_host_port>**，请指定用于提供内容的端口。

   c. 导出本地存储库名称：

```
$ LOCAL_REPOSITORY='<local_repository_name>'
```

   对于 **<local_repository_name>**，请指定要在 registry 中创建的仓库名称，如 **ocp4/openshift4**。

d. 导出要进行镜像的存储库名称：

```
$ PRODUCT_REPO='openshift-release-dev'
```

对于生产环境版本，必须指定 **openshift-release-dev**。

e. 导出 registry pull secret 的路径：

```
$ LOCAL_SECRET_JSON='<path_to_pull_secret>'
```

对于 **<path_to_pull_secret>**，请指定您创建的镜像 registry 的 pull secret 的绝对路径和文件名。

f. 导出发行版本镜像：

```
$ RELEASE_NAME="ocp-release"
```

对于生产环境版本，您必须指定 **ocp-release**。

g. 为您的服务器导出构架类型，如 **x86_64**。

```
$ ARCHITECTURE=<server_architecture>
```

h. 导出托管镜像的目录的路径：

```
$ REMOVABLE_MEDIA_PATH=<path> 1
```

1 指定完整路径，包括开始的前斜杠(/)字符。

3. 将版本镜像(mirror)到镜像 registry：

- 如果您的镜像主机无法访问互联网，请执行以下操作：

i. 将可移动介质连接到连接到互联网的系统。

ii. 查看要镜像的镜像和配置清单：

```
$ oc adm release mirror -a ${LOCAL_SECRET_JSON}  \
    --from=quay.io/${PRODUCT_REPO}/${RELEASE_NAME}:${OCP_RELEASE}-
${ARCHITECTURE} \
    --to=${LOCAL_REGISTRY}/${LOCAL_REPOSITORY} \
    --to-release-
image=${LOCAL_REGISTRY}/${LOCAL_REPOSITORY}:${OCP_RELEASE}-
${ARCHITECTURE} --dry-run
```

iii. 记录上一命令输出中的 **imageContentSources** 部分。您的镜像信息与您的镜像存储库相对应，您必须在安装过程中将 **imageContentSources** 部分添加到 **install-config.yaml** 文件中。

iv. 将镜像镜像到可移动介质的目录中：

```
$ oc adm release mirror -a ${LOCAL_SECRET_JSON} --to-
dir=${REMOVABLE_MEDIA_PATH}/mirror
quay.io/${PRODUCT_REPO}/${RELEASE_NAME}:${OCP_RELEASE}-
```

```
${ARCHITECTURE}
```

v. 将介质上传到受限网络环境中，并将镜像上传到本地容器 registry。

```
$ oc image mirror -a ${LOCAL_SECRET_JSON} --from-
dir=${REMOVABLE_MEDIA_PATH}/mirror
"file://openshift/release:${OCP_RELEASE}*"
${LOCAL_REGISTRY}/${LOCAL_REPOSITORY} 1
```

1 对于 **REMOVABLE_MEDIA_PATH**，您必须使用与镜像镜像时指定的同一路径。

- 如果本地容器 registry 连接到镜像主机，请执行以下操作：

i. 使用以下命令直接将发行版镜像推送到本地 registry:

```
$ oc adm release mirror -a ${LOCAL_SECRET_JSON}  \
    --from=quay.io/${PRODUCT_REPO}/${RELEASE_NAME}:${OCP_RELEASE}-
${ARCHITECTURE} \
    --to=${LOCAL_REGISTRY}/${LOCAL_REPOSITORY} \
    --to-release-
image=${LOCAL_REGISTRY}/${LOCAL_REPOSITORY}:${OCP_RELEASE}-
${ARCHITECTURE}
```

该命令将发行信息提取为摘要，其输出包括安装集群时所需的 **imageContentSources** 数据。

ii. 记录上一命令输出中的 **imageContentSources** 部分。您的镜像信息与您的镜像存储库相对应，您必须在安装过程中将 **imageContentSources** 部分添加到 **install-config.yaml** 文件中。

注意

镜像名称在镜像过程中被修补到 Quay.io， podman 镜像将在 bootstrap 虚拟机的 registry 中显示 Quay.io。

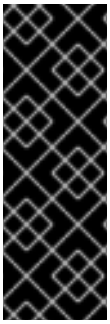
4. 要创建基于您镜像内容的安装程序，请提取内容并将其固定到发行版中：

- 如果您的镜像主机无法访问互联网，运行以下命令：

```
$ oc adm release extract -a ${LOCAL_SECRET_JSON} --command=openshift-install
"${LOCAL_REGISTRY}/${LOCAL_REPOSITORY}:${OCP_RELEASE}"
```

- 如果本地容器 registry 连接到镜像主机，请运行以下命令：

```
$ oc adm release extract -a ${LOCAL_SECRET_JSON} --command=openshift-install
"${LOCAL_REGISTRY}/${LOCAL_REPOSITORY}:${OCP_RELEASE}-
${ARCHITECTURE}"
```



重要

要确保将正确的镜像用于您选择的 OpenShift Container Platform 版本，您必须从镜像内容中提取安装程序。

您必须在有活跃互联网连接的机器上执行这个步骤。

如果您位于断开连接的环境中,请使用 **--image** 标志作为 must-gather 的一部分，指向有效负载镜像。

1.8. 在断开连接的环境中的 CLUSTER SAMPLES OPERATOR

在断开连接的环境中，在安装集群后执行额外的步骤来配置 Cluster Samples Operator。

1.9. 后续步骤

- Mirror 您要在集群中安装的 Operator 的 OperatorHub 镜像。
- 在您在受限网络中置备的基础架构上安装集群，如 VMware vSphere、裸机或 Amazon Web Services。

1.10. 其他资源

- 有关使用 must-gather 的更多信息，请参阅收集有关特定功能的数据。

第 2 章 在 AWS 上安装

2.1. 配置 AWS 帐户

在安装 OpenShift Container Platform 之前，您必须先配置 Amazon Web Services（AWS）帐户。

2.1.1. 配置路由 53（Route 53）

要安装 OpenShift Container Platform，您使用的 Amazon Web Services (AWS) 帐户必须在 Route 53 服务中有一个专用的公共托管区。此区域必须对域具有权威。Route 53 服务为集群外部连接提供集群 DNS 解析和名称查询。

流程

1. 标识您的域或子域，以及注册商（registrar）。您可以转移现有的域和注册商，或通过 AWS 或其他来源获取新的域和注册商。

注意

如果您通过 AWS 购买了一个新域，则需要一定时间来传播相关的 DNS 更改信息。有关通过 AWS 购买域的更多信息，请参阅 AWS 文档中的使用 Amazon Route 53 注册域名。

2. 如果您使用现有的域和注册商，请将其 DNS 迁移到 AWS。请参阅 AWS 文档中的使 Amazon Route 53 成为现有域的 DNS 服务。

3. 为您的域或子域创建一个公共托管区。请参阅 AWS 文档中的创建公共托管区。
使用合适的根域（如 **openshiftcorp.com**）或子域（如 **clusters.openshiftcorp.com**）。

4. 从托管区记录中提取新的权威名称服务器。请参阅 AWS 文档中的获取公共托管区的名称服务器。

5. 更新域所用 AWS Route 53 名称服务器的注册商记录。例如，如果您将域注册到不同帐户中的 Route 53 服务，请参阅 AWS 文档中的以下主题：添加或更改名称服务器或粘附记录。

6. 如果使用子域，请将其委托记录添加到父域中。这为子域赋予 Amazon Route 53 责任。按照父域的 DNS 供应商概述的委托程序。请参阅 创建使用 Amazon Route 53 作为 DNS 服务的子域，而无需迁移 AWS 文档 中的父域以获取示例高级流程。

2.1.1.1. AWS Route 53 的 Ingress Operator 端点配置

如果您在 Amazon Web Services（AWS）GovCloud(US)US-West 或 US-East 区域中安装，Ingress Operator 使用 **us-gov-west-1** 区域用于 Route53 并标记 API 客户端。

如果配置了带有字符串 'us-gov-east-1' 的自定义端点，Ingress Operator 使用 **https://tagging.us-gov-west-1.amazonaws.com** 作为 tagging API 端点。

有关 AWS GovCloud（US）端点的更多信息，请参阅 AWS 文档中的有关 GovCloud(US)的服务端点的内容。

重要

在 **us-gov-east-1** 区域中安装时，AWS GovCloud 不支持私有的、断开连接的安装。

Route 53 配置示例

```
platform:
  aws:
    region: us-gov-west-1
    serviceEndpoints:
    - name: ec2
      url: https://ec2.us-gov-west-1.amazonaws.com
    - name: elasticloadbalancing
      url: https://elasticloadbalancing.us-gov-west-1.amazonaws.com
    - name: route53
      url: https://route53.us-gov.amazonaws.com 1
    - name: tagging
      url: https://tagging.us-gov-west-1.amazonaws.com 2
```

1 对于所有两个 AWS GovCloud(US)区域，Route53 默认为 **https://route53.us-gov.amazonaws.com**。

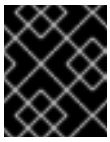
2 只有 US-West 区域有标记端点。如果集群位于另一个区域，则省略此参数。

2.1.2. AWS 帐户限值

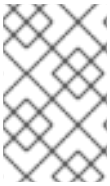
OpenShift Container Platform 集群使用诸多 Amazon Web Services (AWS) 组件，默认的服务限值会影响您安装 OpenShift Container Platform 集群的能力。如果您使用特定的集群配置，在某些 AWS 区域部署集群，或者从您的帐户运行多个集群，您可能需要为 AWS 帐户请求其他资源。

下表总结了 AWS 组件，它们的限值可能会影响您安装和运行 OpenShift Container Platform 集群的能力。

组件	默认可用的集群数	默认 AWS 限值	描述
实例限值	可变	可变	默认情况下，每个集群创建以下实例： • 一台 Bootstrap 机器，在安装后删除 • 三个 control plane 节点（也称为 master 节点） • 三个 worker 节点 这些实例类型数量在新帐户的默认限值之内。若要部署更多 worker 节点、启用自动扩展、部署大型工作负载或使用不同的实例类型，请检查您的帐户限制，以确保集群可以部署您需要的机器。 在大多数区域中，bootstrap 和 worker 机器使用 **m4.large** 机器，control plane 机器使用 **m4.xlarge** 实例。在一些区域，包括所有不支持这些实例类型的区域，则使用 **m5.large** 和 **m5.xlarge** 实例。

组件	默认可用的集群数	默认 AWS 限值	描述
弹性 IP (EIP)	0 到 1	每个帐户 5 个 EIP	要在高可用性配置中置备集群，安装程序将为区域中的每个可用区创建一个公共和专用子网。每个专用子网都需要 NAT 网关，每个 NAT 网关需要单独的弹性 IP。查看 AWS 区域图来确定每个区域有多少个可用区。要利用默认高可用性，请在至少含有三个可用区的区域安装集群。要在有超过五个可用区的区域安装集群，您必须提高 EIP 限值。  **重要** 要使用 **us-east-1** 区域，必须提高您帐户的 EIP 限值。
虚拟私有云 (VPC)	5	每个区域 5 个 VPC	每个集群创建自己的 VPC。
弹性负载均衡 (ELB/NLB)	3	每个区域 20 个	在默认情况下，每个集群为 master API 服务器创建一个内部和外部网络负载均衡器，并为路由器创建一个典型的弹性负载均衡器。使用类型 **LoadBalancer** 部署更多 Kubernetes **Service** 对象将创建额外的负载均衡器。
NAT 网关	5	每个可用区 5 个	集群在每个可用区中部署一个 NAT 网关。
弹性网络接口 (ENI)	至少 12 个	每个区域 350 个	默认安装创建 21 个 ENI，并为区域中的每个可用区创建一个 ENI。例如，**us-east-1** 区域包含六个可用区，因此在该区部署的集群将使用 27 个 ENI。查看 AWS 区域图来确定每个区域有多少个可用区。 针对根据集群使用情况和部署的工作负载创建的额外机器和弹性负载均衡器，为其创建额外的 ENI。
VPC 网关	20	每个帐户 20 个	每个集群创建一个 VPC 网关来访问 S3。
S3 存储桶	99	每个帐户有 100 个存储桶	因为安装过程会创建一个临时存储桶，并且每个集群中的 registry 组件会创建一个存储桶，所以您只能为每个 AWS 帐户创建 99 个 OpenShift Container Platform 集群。
安全组	250	每个帐户 2,500 个	每个集群创建 10 个不同的安全组。

2.1.3. 所需的 AWS 权限



注意

您的 IAM 用户必须具有 region **us-east-1** s 中的 permissions **tag:GetResources** 才能删除基本集群资源。作为 AWS API 要求的一部分，OpenShift Container Platform 安装程序在此区域中执行各种操作。

将 **AdministratorAccess** 策略附加到您在 Amazon Web Services (AWS) 中创建的 IAM 用户时，授予该用户所有需要的权限。要部署 OpenShift Container Platform 集群的所有组件，IAM 用户需要以下权限：

例 2.1. 安装所需的 EC2 权限

- **tag:TagResources**
- **tag:UntagResources**
- **ec2:AllocateAddress**
- **ec2:AssociateAddress**
- **ec2:AuthorizeSecurityGroupEgress**
- **ec2:AuthorizeSecurityGroupIngress**
- **ec2:CopyImage**
- **ec2:CreateNetworkInterface**
- **ec2:AttachNetworkInterface**
- **ec2:CreateSecurityGroup**
- **ec2:CreateTags**
- **ec2:CreateVolume**
- **ec2:DeleteSecurityGroup**
- **ec2:DeleteSnapshot**
- **ec2:DeleteTags**
- **ec2:DeregisterImage**
- **ec2:DescribeAccountAttributes**
- **ec2:DescribeAddresses**
- **ec2:DescribeAvailabilityZones**
- **ec2:DescribeDhcpOptions**
- **ec2:DescribeImages**
- **ec2:DescribeInstanceAttribute**
- **ec2:DescribeInstanceCreditSpecifications**

- **ec2:DescribeInstances**
- **ec2:DescribeInternetGateways**
- **ec2:DescribeKeyPairs**
- **ec2:DescribeNatGateways**
- **ec2:DescribeNetworkAcls**
- **ec2:DescribeNetworkInterfaces**
- **ec2:DescribePrefixLists**
- **ec2:DescribeRegions**
- **ec2:DescribeRouteTables**
- **ec2:DescribeSecurityGroups**
- **ec2:DescribeSubnets**
- **ec2:DescribeTags**
- **ec2:DescribeVolumes**
- **ec2:DescribeVpcAttribute**
- **ec2:DescribeVpcClassicLink**
- **ec2:DescribeVpcClassicLinkDnsSupport**
- **ec2:DescribeVpcEndpoints**
- **ec2:DescribeVpcs**
- **ec2:GetEbsDefaultKmsKeyId**
- **ec2:ModifyInstanceAttribute**
- **ec2:ModifyNetworkInterfaceAttribute**
- **ec2:ReleaseAddress**
- **ec2:RevokeSecurityGroupEgress**
- **ec2:RevokeSecurityGroupIngress**
- **ec2:RunInstances**
- **ec2:TerminateInstances**

例 2.2. 安装过程中创建网络资源所需的权限

- **ec2:AssociateDhcpOptions**
- **ec2:AssociateRouteTable**

- **ec2:AttachInternetGateway**
- **ec2:CreateDhcpOptions**
- **ec2:CreateInternetGateway**
- **ec2:CreateNatGateway**
- **ec2:CreateRoute**
- **ec2:CreateRouteTable**
- **ec2:CreateSubnet**
- **ec2:CreateVpc**
- **ec2:CreateVpcEndpoint**
- **ec2:ModifySubnetAttribute**
- **ec2:ModifyVpcAttribute**

注意

如果您使用现有的 VPC，您的帐户不需要这些权限来创建网络资源。

例 2.3. 安装所需的 Elastic Load Balancing 权限(ELB)

- **elasticloadbalancing:AddTags**
- **elasticloadbalancing:ApplySecurityGroupsToLoadBalancer**
- **elasticloadbalancing:AttachLoadBalancerToSubnets**
- **elasticloadbalancing:ConfigureHealthCheck**
- **elasticloadbalancing:CreateLoadBalancer**
- **elasticloadbalancing:CreateLoadBalancerListeners**
- **elasticloadbalancing:DeleteLoadBalancer**
- **elasticloadbalancing:DeregisterInstancesFromLoadBalancer**
- **elasticloadbalancing:DescribeInstanceHealth**
- **elasticloadbalancing:DescribeLoadBalancerAttributes**
- **elasticloadbalancing:DescribeLoadBalancers**
- **elasticloadbalancing:DescribeTags**
- **elasticloadbalancing:ModifyLoadBalancerAttributes**
- **elasticloadbalancing:RegisterInstancesWithLoadBalancer**

- **elasticloadbalancing:SetLoadBalancerPoliciesOfListener**

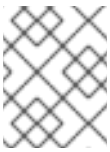
例 2.4. 安装所需的 Elastic Load Balancing 权限(ELBv2)

- **elasticloadbalancing:AddTags**
- **elasticloadbalancing:CreateListener**
- **elasticloadbalancing:CreateLoadBalancer**
- **elasticloadbalancing:CreateTargetGroup**
- **elasticloadbalancing:DeleteLoadBalancer**
- **elasticloadbalancing:DeregisterTargets**
- **elasticloadbalancing:DescribeListeners**
- **elasticloadbalancing:DescribeLoadBalancerAttributes**
- **elasticloadbalancing:DescribeLoadBalancers**
- **elasticloadbalancing:DescribeTargetGroupAttributes**
- **elasticloadbalancing:DescribeTargetHealth**
- **elasticloadbalancing:ModifyLoadBalancerAttributes**
- **elasticloadbalancing:ModifyTargetGroup**
- **elasticloadbalancing:ModifyTargetGroupAttributes**
- **elasticloadbalancing:RegisterTargets**

例 2.5. 安装所需的 IAM 权限

- **iam:AddRoleToInstanceProfile**
- **iam:CreateInstanceProfile**
- **iam:CreateRole**
- **iam:DeleteInstanceProfile**
- **iam:DeleteRole**
- **iam:DeleteRolePolicy**
- **iam:GetInstanceProfile**
- **iam:GetRole**
- **iam:GetRolePolicy**
- **iam:GetUser**

- **iam:ListInstanceProfilesForRole**
- **iam:ListRoles**
- **iam:ListUsers**
- **iam:PassRole**
- **iam:PutRolePolicy**
- **iam:RemoveRoleFromInstanceProfile**
- **iam:SimulatePrincipalPolicy**
- **iam:TagRole**



注意

如果您还没有在 AWS 帐户中创建弹性负载均衡器（ELB），IAM 用户还需要 **iam:CreateServiceLinkedRole** 权限。

例 2.6. 安装所需的 Route 53 权限

- **route53:ChangeResourceRecordSets**
- **route53:ChangeTagsForResource**
- **route53:CreateHostedZone**
- **route53:DeleteHostedZone**
- **route53:GetChange**
- **route53:GetHostedZone**
- **route53:ListHostedZones**
- **route53:ListHostedZonesByName**
- **route53:ListResourceRecordSets**
- **route53:ListTagsForResource**
- **route53:UpdateHostedZoneComment**

例 2.7. 安装所需的 S3 权限

- **s3:CreateBucket**
- **s3:DeleteBucket**
- **s3:GetAccelerateConfiguration**
- **s3:GetBucketAcl**

- **s3:GetBucketCors**
- **s3:GetBucketLocation**
- **s3:GetBucketLogging**
- **s3:GetBucketObjectLockConfiguration**
- **s3:GetBucketReplication**
- **s3:GetBucketRequestPayment**
- **s3:GetBucketTagging**
- **s3:GetBucketVersioning**
- **s3:GetBucketWebsite**
- **s3:GetEncryptionConfiguration**
- **s3:GetLifecycleConfiguration**
- **s3:GetReplicationConfiguration**
- **s3:ListBucket**
- **s3:PutBucketAcl**
- **s3:PutBucketTagging**
- **s3:PutEncryptionConfiguration**

例 2.8. 集群 Operators 所需的 S3 权限

- **s3:DeleteObject**
- **s3:GetObject**
- **s3:GetObjectAcl**
- **s3:GetObjectTagging**
- **s3:GetObjectVersion**
- **s3:PutObject**
- **s3:PutObjectAcl**
- **s3:PutObjectTagging**

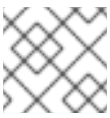
例 2.9. 删除基本集群资源所需的权限

- **autoscaling:DescribeAutoScalingGroups**
- **ec2:DeleteNetworkInterface**

- **ec2:DeleteVolume**
- **elasticloadbalancing:DeleteTargetGroup**
- **elasticloadbalancing:DescribeTargetGroups**
- **iam:DeleteAccessKey**
- **iam:DeleteUser**
- **iam:ListAttachedRolePolicies**
- **iam:ListInstanceProfiles**
- **iam:ListRolePolicies**
- **iam:ListUserPolicies**
- **s3:DeleteObject**
- **s3:ListBucketVersions**
- **tag:GetResources**

例 2.10. 删除网络资源所需的权限

- **ec2:DeleteDhcpOptions**
- **ec2:DeleteInternetGateway**
- **ec2:DeleteNatGateway**
- **ec2:DeleteRoute**
- **ec2:DeleteRouteTable**
- **ec2:DeleteSubnet**
- **ec2:DeleteVpc**
- **ec2:DeleteVpcEndpoints**
- **ec2:DetachInternetGateway**
- **ec2:DisassociateRouteTable**
- **ec2:ReplaceRouteTableAssociation**



注意

如果您使用现有的 VPC，您的帐户不需要这些权限来删除网络资源。

例 2.11. 创建清单所需的额外 IAM 和 S3 权限

- **iam:DeleteAccessKey**
- **iam:DeleteUser**
- **iam:DeleteUserPolicy**
- **iam:GetUserPolicy**
- **iam:ListAccessKeys**
- **iam:PutUserPolicy**
- **iam:TagUser**
- **iam:GetUserPolicy**
- **iam:ListAccessKeys**
- **s3:PutBucketPublicAccessBlock**
- **s3:GetBucketPublicAccessBlock**
- **s3:PutLifecycleConfiguration**
- **s3:HeadBucket**
- **s3:ListBucketMultipartUploads**
- **s3:AbortMultipartUpload**

注意

如果您要使用 mint 模式管理云供应商凭证，IAM 用户还需要 The **iam:CreateAccessKey and iam:CreateUser** 权限。

例 2.12. 安装时配额检查的可选权限

- **servicequotas:ListAWSDefaultServiceQuotas**

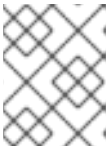
2.1.4. 创建 IAM 用户

每个 Amazon Web Services (AWS) 帐户都包含一个根用户帐户，它基于您用来创建帐户的电子邮件地址。这是一个高权限帐户，建议仅用于初始帐户和账单配置、创建初始用户集，以及保护帐户安全。

在安装 OpenShift Container Platform 之前，请创建一个辅助 IAM 管理用户。完成 AWS 文档中所述的在 AWS 帐户中创建 IAM 用户流程时，请设置以下选项：

流程

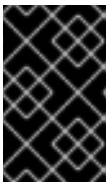
1. 指定 IAM 用户名并选择 **Programmatic access**。
2. 附加 **AdministratorAccess** 策略，以确保帐户有充足的权限来创建集群。此策略让集群能够为每个 OpenShift Container Platform 组件授予凭证。集群只为组件授予它们需要的凭证。



注意

虽然可以创建赋予所有所需 AWS 权限的策略并将其附加到用户，但这不是首选的选项。集群将无法为各个组件授予额外的凭证，因此所有组件都使用相同的凭证。

3. 可选：通过附加标签向用户添加元数据。
4. 确认您指定的用户名被授予了 **AdministratorAccess** 策略。
5. 记录访问密钥 ID 和 Secret 访问密钥值。在配置本地机器时，您必须使用这些值来运行安装程序。



重要

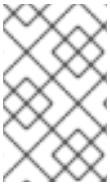
在部署集群时，您无法在使用多因素验证设备来验证 AWS 的同时使用您生成的临时会话令牌。在集群的整个生命周期中，集群会持续使用您的当前 AWS 凭证来创建 AWS 资源，因此您必须使用基于密钥的长期凭证。

其他资源

- 有关在安装前将 Cloud Credential Operator（CCO）设置为手动模式的步骤，请参阅手动为 AWS 创建 IAM。在无法使用云身份和访问管理（IAM）API 的环境里，或不希望将管理员级别的凭证 secret 保存在集群 **kube-system** 项目中时，可以使用这个模式。

2.1.5. 支持的 AWS 区域

您可以将 OpenShift Container Platform 集群部署到以下公共区域：



注意

您的 IAM 用户必须具有 region **us-east-1** s 中的 permissions **tag:GetResources** 才能删除基本集群资源。作为 AWS API 要求的一部分，OpenShift Container Platform 安装程序在此区域中执行各种操作。

- **af-south-1** (Cape Town)
- **ap-east-1** (Hong Kong)
- **ap-northeast-1** (Tokyo)
- **ap-northeast-2** (Seoul)
- **ap-northeast-3** (Osaka)
- **ap-south-1** (Mumbai)
- **ap-southeast-1** (Singapore)
- **ap-southeast-2** (Sydney)
- **ca-central-1** (Central)
- **eu-central-1** (Frankfurt)
- **eu-north-1** (Stockholm)

- **eu-south-1** (Milan)
- **eu-west-1** (Ireland)
- **eu-west-2** (London)
- **eu-west-3** (Paris)
- **me-south-1** (Bahrain)
- **sa-east-1** (São Paulo)
- **us-east-1** (N. Virginia)
- **us-east-2** (Ohio)
- **us-west-1** (N. California)
- **us-west-2** (Oregon)

支持以下 AWS GovCloud 区域：

- **us-gov-west-1**
- **us-gov-east-1**

2.1.6. 后续步骤

- 安装 OpenShift Container Platform 集群：
  - 使用安装程序置备 基础架构默认选项快速安装集群
  - 在安装程序置备的基础架构中使用云自定义安装集群
  - 使用网络自定义在安装程序置备的基础架构上安装集群
  - 使用 CloudFormation 模板在 AWS 中用户置备的基础架构上安装集群

2.2. 为 AWS 手动创建 IAM

在无法访问云身份和访问管理（IAM）API 的环境中，或者管理员更不希望将管理员级别的凭证 secret 存储在集群 **kube-system** 命名空间中时，可以在安装前将 Cloud Credential Operator（CCO）放入手动模式。

2.2.1. 在 kube-system 项目中存储管理员级别的 secret 的替代方案

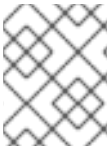
Cloud Credential Operator（CCO） 将云供应商凭证作为 Kubernetes 自定义资源定义 (CRD) 进行管理。您可以通过在 **install-config.yaml** 文件中为 **credentialsMode** 参数设置不同的值，来配置 CCO 来满足机构的安全要求。

如果您不希望在集群 **kube-system** 项目中存储管理员级别的凭证 secret，您可以在安装 OpenShift Container Platform 时选择以下选项之一：

- **手动管理云凭证**：
  您可以将 CCO 的 **credentialsMode** 参数设置为 **Manual** 以手动管理云凭证。使用手动模式可允许每个集群组件只拥有所需的权限，而无需在集群中存储管理员级别的凭证。如果您的环境没有

连接到云供应商公共 IAM 端点，您还可以使用此模式。但是，每次升级都必须手动将权限与新发行镜像协调。您还必须手动为每个请求它们的组件提供凭证。

- **使用 mint 模式安装 OpenShift Container Platform 后删除管理员级别的凭证 secret:** 如果您使用 CCO，并将 **credentialsMode** 参数设置为 **Mint**，您可以在安装 OpenShift Container Platform 后删除或轮转管理员级别的凭证。Mint 模式是 CCO 的默认配置。这个选项需要在安装过程中存在管理员级别的凭证。在安装过程中使用管理员级别的凭证来模拟授予某些权限的其他凭证。原始凭证 secret 不会永久存储在集群中。



注意

在非 z-stream 升级前，您必须使用管理员级别的凭证重新恢复凭证 secret。如果没有凭证，则可能无法进行升级。

其他资源

- 要了解如何在安装 OpenShift Container Platform 后轮转或删除管理员级别的凭证 secret，请参阅轮转或删除云供应商凭证。
- 有关所有可用 CCO 凭证模式及其支持的平台的详情，请参阅 Cloud Credential Operator。

2.2.2. 手动创建 IAM

在无法访问云身份和访问管理（IAM）API 的环境中，或者管理员更不希望将管理员级别的凭证 secret 存储在集群 **kube-system** 命名空间中时，可以在安装前将 Cloud Credential Operator（CCO）放入手动模式。

流程

1. 切换到包含安装程序的目录并创建 **install-config.yaml** 文件：

```
$ openshift-install create install-config --dir <installation_directory>
```

2. 编辑 **install-config.yaml** 配置文件，把其中的 **credentialsMode** 参数设置为**Manual**。

示例 install-config.yaml 配置文件

```
apiVersion: v1
baseDomain: cluster1.example.com
credentialsMode: Manual (1)
compute:
- architecture: amd64
  hyperthreading: Enabled
...
```

(1) 添加这一行将 **credentialsMode** 参数设置为 **Manual**。

3. 要生成清单，请在包含安装程序的目录中运行以下命令：

```
$ openshift-install create manifests --dir <installation_directory> (1)
```

(1) 对于 **<installation_directory>**，请指定用于保存安装程序所创建的文件的目录名称。

4. 删除使用本地云凭证创建的 **admin** 凭证 secret。这会防止您的 **admin** 凭证存储在集群中：

```
$ rm mycluster/openshift/99_cloud-creds-secret.yaml
```

5. 从包含安装程序的目录中，获取 **openshift-install** 二进制文件要使用的 OpenShift Container Platform 发行镜像详情：

```
$ openshift-install version
```

输出示例

```
release image quay.io/openshift-release-dev/ocp-release:4.y.z-x86_64
```

6. 针对您要部署到的云，找到此发行版本镜像中的所有 **CredentialsRequests** 对象：

```
$ oc adm release extract quay.io/openshift-release-dev/ocp-release:4.y.z-x86_64 --
credentials-requests --cloud=aws
```

这会显示每个请求的详情。

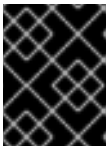
CredentialsRequest 对象示例

```
apiVersion: cloudcredential.openshift.io/v1
kind: CredentialsRequest
metadata:
  name: cloud-credential-operator-iam-ro
  namespace: openshift-cloud-credential-operator
spec:
  secretRef:
    name: cloud-credential-operator-iam-ro-creds
    namespace: openshift-cloud-credential-operator
  providerSpec:
    apiVersion: cloudcredential.openshift.io/v1
    kind: AWSProviderSpec
    statementEntries:
    - effect: Allow
      action:
      - iam:GetUser
      - iam:GetUserPolicy
      - iam:ListAccessKeys
      resource: "*"
```

7. 在之前生成的 **openshift-install** 清单目录中为 secret 创建 YAML 文件。secret 必须使用在 **spec.secretRef** 中为每个 **credentialsRequest** 定义的命名空间和 secret 名称存储 。secret 数据的格式因云供应商而异。

8. 从包含安装程序的目录中，开始创建集群：

```
$ openshift-install create cluster --dir <installation_directory>
```



重要

在升级使用手动维护凭证的集群前，必须确保 CCO 处于可升级状态。详情请参阅您的云供应商的*对手动维护凭证的集群进行升级*部分的内容。

2.2.3. 管理凭证 root secret 格式

每个云供应商都使用 **kube-system** 命名空间中的一个凭证 root secret，用于满足所有凭证请求并创建它们相应的 secret。这可以通过 mint 新凭证（*mint mode*），或复制凭证 root secret（*passthrough mode*）实现。

secret 的格式因云而异，也用于每个 **CredentialsRequest** secret。

Amazon Web Services(AWS)secret 格式

```
apiVersion: v1
kind: Secret
metadata:
  namespace: kube-system
  name: aws-creds
stringData:
  aws_access_key_id: <AccessKeyID>
  aws_secret_access_key: <SecretAccessKey>
```

2.2.4. 使用手动维护的凭证升级集群

如果在未来的发行版本中添加了凭证，则使用手动维护凭证的集群的 Cloud Credential Operator（CCO）**可升级**状态会变为 **false**。对于次版本（例如从 4.5 到 4.6），这个状态会阻止升级，直到解决了更新的权限。对于 z-stream 版本（例如从 4.5.10 到 4.5.11），升级不会受阻，但必须为新版本更新凭证。

使用 Web 控制台的 **Administrator** 视角来判断 CCO 是否可以升级。

1. 导航至 **Administration → Cluster Settings**。
2. 要查看 CCO 状态详情，请点 **Cluster Operators** 列表中的 **cloud-credential**。
3. 如果 **Conditions** 部分中的 **Upgradeable** 状态为 **False**，请检查新发行版本的 **credentialsRequests**，并在升级前更新集群中手动维护的凭证以匹配。

除了为您要升级到的发行版本镜像创建新凭证外，还需要查看现有凭证所需的权限，并满足新发行版本中现有组件的所有新权限要求。CCO 无法检测到这些不匹配的问题，且在此情况下无法将 **upgradable** 设置为 **false**。

详情请参阅您的云供应商的*手动创建 IAM* 部分来了解如何获取和使用您的云所需的凭证。

2.2.5. Mint 模式

Mint 模式是 OpenShift Container Platform 的默认和推荐的 Cloud Credential Operator（CCO）凭证模式。在这种模式中，CCO 使用提供的管理员级云凭证来运行集群。AWS、GCP 和 Azure 支持 Mint 模式。

在 mint 模式中，**admin** 凭证存储在 **kube-system** 命名空间中，然后由 CCO 使用来处理集群中的 **CredentialsRequest** 对象，并为每个对象创建具有特定权限的用户。

mint 模式的好处包括：

- 每个集群组件只有其所需权限
- 云凭证的自动、持续协调，包括升级可能需要的额外凭证或权限

mint 模式的一个缺陷是，**admin** 凭证需要存储在集群 **kube-system** 的 secret 中。

2.2.6. 带有删除或轮转管理员凭证的 Mint 模式

目前，只有 AWS 支持这个模式。

在这个模式中，用户使用类似正常的 mint 模式的 **admin** 凭证安装 OpenShift Container Platform。但是，此模式会在集群安装后删除 **admin** 凭证 secret。

管理员可以让 Cloud Credential Operator 自行请求只读凭证，许它验证所有 **CredentialsRequest** 对象是否有其所需的权限。因此，除非需要更改内容，否则不需要 **admin** 凭证。删除关联的凭证后，可以根据需要在底层云上销毁它。

在升级前，应该恢复 **admin** 凭证。以后，如果凭证不存在，升级可能会阻止。

admin 凭证不会永久存储在集群中。

这个模式仍然需要在一个短的时间内，集群中存在 **admin** 凭证。它还需要为每个升级使用 **admin** 凭证手动重新生成 secret。

2.2.7. 后续步骤

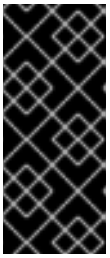
- 安装 OpenShift Container Platform 集群：
  - 使用安装程序置备的基础架构默认选项在 AWS 上快速安装集群
  - 在安装程序置备的基础架构中使用云自定义安装集群
  - 使用网络自定义在安装程序置备的基础架构上安装集群
  - 使用 CloudFormation 模板在 AWS 中用户置备的基础架构上安装集群

2.3. 在 AWS 上快速安装集群

在 OpenShift Container Platform 版本 4.6 中，您可以使用默认配置选项在 Amazon Web Services（AWS）上安装集群。

2.3.1. 先决条件

- 查看有关 OpenShift Container Platform 安装和更新流程的详细信息。
- 配置 AWS 帐户以托管集群。



重要

如果您的计算机上存储有 AWS 配置集，则不要在使用多因素验证设备的同时使用您生成的临时会话令牌。在集群的整个生命周期中，集群会持续使用您的当前 AWS 凭证来创建 AWS 资源，因此您必须使用基于密钥的长期凭证。要生成适当的密钥，请参阅 AWS 文档中的管理 IAM 用户的访问密钥。您可在运行安装程序时提供密钥。

- 如果使用防火墙，则必须将其配置为允许集群需要访问的站点。
- 如果不允许系统管理身份和访问管理（IAM），集群管理员可以 手动创建和维护 IAM 凭证。手动模式也可以用于云 IAM API 无法访问的环境中。

2.3.2. OpenShift Container Platform 的互联网访问

在 OpenShift Container Platform 4.6 中，您需要访问互联网来安装集群。

您必须具有以下互联网访问权限：

- 访问 OpenShift Cluster Manager 以下载安装程序并执行订阅管理。如果集群可以访问互联网，并且没有禁用 Telemetry，该服务会自动授权您的集群。
- 访问 Quay.io，以获取安装集群所需的软件包。
- 获取执行集群更新所需的软件包。

重要

如果您的集群无法直接访问互联网，则可以在置备的某些类基础架构上执行受限网络安装。在此过程中，您要下载所需的内容，并使用它在镜像 registry（mirror registry） 中填充安装集群并生成安装程序所需的软件包。对于某些安装类型，集群要安装到的环境不需要访问互联网。在更新集群之前，要更新 registry 镜像系统中的内容。

2.3.3. 生成 SSH 私钥并将其添加到代理中

如果要在集群上执行安装调试或灾难恢复，则必须为 **ssh-agent** 和安装程序提供 SSH 密钥。您可以使用此密钥访问公共集群中的 bootstrap 机器来排除安装问题。

注意

在生产环境中，您需要进行灾难恢复和调试。

您可以使用此密钥以 **core** 用户身份通过 SSH 连接到 master 节点。在部署集群时，此密钥会添加到 **core** 用户的 **~/.ssh/authorized_keys** 列表中。

注意

您必须使用一个本地密钥，而不要使用在特定平台上配置的密钥，如 AWS 密钥对。

流程

1. 如果还没有为计算机上免密码身份验证而配置的 SSH 密钥，请创建一个。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ ssh-keygen -t ed25519 -N '' \
  -f <path>/<file_name> 1
```

1 指定新 SSH 密钥的路径和文件名，如 **~/.ssh/id_rsa**。如果您已有密钥对，请确保您的公钥位于 **~/.ssh** 目录中。

运行此命令会在指定的位置生成不需要密码的 SSH 密钥。

注意

如果您计划在 **x86_64** 架构中安装使用 FIPS 验证的/Modules in Process 加密库的 OpenShift Container Platform 集群，不要创建使用 **ed25519** 算法的密钥。反之，创建一个使用 **rsa** 或 **ecdsa** 算法的密钥。

2. 作为后台任务启动 **ssh-agent** 进程：

```
$ eval "$(ssh-agent -s)"
```

输出示例

```
Agent pid 31874
```

注意

如果您的集群采用 FIPS 模式，则只使用 FIPS 兼容算法来生成 SSH 密钥。密钥必须是 RSA 或 ECDSA。

3. 将 SSH 私钥添加到 **ssh-agent**：

```
$ ssh-add <path>/<file_name> 1
```

输出示例

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

1 指定 SSH 私钥的路径和文件名，如 **~/.ssh/id_rsa**

后续步骤

- 在安装 OpenShift Container Platform 时，为安装程序提供 SSH 公钥。

2.3.4. 获取安装程序

在安装 OpenShift Container Platform 之前，将安装文件下载到本地计算机上。

先决条件

- 运行 Linux 或 macOS 的计算机，本地磁盘空间为 500 MB

流程

1. 访问 OpenShift Cluster Manager 站点的 Infrastructure Provider 页面。如果您有红帽帐号，请使用自己的凭证登录。如果没有，请创建一个帐户。
2. 选择您的基础架构供应商。
3. 进入适用于您的安装类型的页面，下载您的操作系统的安装程序，并将文件放在要保存安装配置文件的目录中。。

重要

安装程序会在用来安装集群的计算机上创建若干文件。在完成集群安装后，您必须保留安装程序和安装程序所创建的文件。这两个文件都需要删除集群。

重要

删除安装程序创建的文件不会删除您的集群，即使集群在安装过程中失败也是如此。要删除集群，为特定云供应商完成 OpenShift Container Platform 卸载流程。

4. 提取安装程序。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ tar xvf openshift-install-linux.tar.gz
```

5. 从 Red Hat OpenShift Cluster Manager 下载安装 pull secret 。通过此 pull secret，您可以进行所含授权机构提供的服务的身份验证，这些服务包括为 OpenShift Container Platform 组件提供容器镜像的 Quay.io。

2.3.5. 部署集群

您可以在兼容云平台中安装 OpenShift Container Platform。

重要

安装程序的 **create cluster** 命令只能在初始安装过程中运行一次。

先决条件

- 配置托管集群的云平台的帐户。
- 获取 OpenShift Container Platform 安装程序以及集群的 pull secret。

流程

1. 更改为包含安装程序的目录并初始化集群部署：

```
$ ./openshift-install create cluster --dir <installation_directory> \ 1
    --log-level=info 2
```

1. 对于 **<installation_directory>**，请指定用于保存安装程序所创建的文件的目录名称。
2. 要查看不同的安装详情，请指定 **warn**、**debug** 或 **error**，而不要指定 **info**。

重要

指定一个空目录。一些安装信息，如 bootstrap X.509 证书，有较短的过期间隔，因此不要重复使用安装目录。如果要重复使用另一个集群安装中的个别文件，可以将其复制到您的目录中。但是，一些安装数据的文件名可能会在发行版本之间有所改变。从 OpenShift Container Platform 老版本中复制安装文件时要格外小心。

在提示符处提供值：

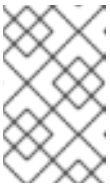
a. 可选：选择用来访问集群机器的 SSH 密钥。

注意

对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 **ssh-agent** 进程使用的 SSH 密钥。

b. 选择 **aws** 作为目标平台。

c. 如果计算机上没有保存 Amazon Web Services (AWS) 配置集，请为您配置用于运行安装程序的用户输入 AWS 访问密钥 ID 和 Secret 访问密钥。



注意

AWS 访问密钥 ID 和 secret 访问密钥存储在安装主机上当前用户主目录中的 **~/.aws/credentials** 中。如果文件中不存在导出的配置集凭证，安装程序会提示您输入凭证。您向安装程序提供的所有凭证都存储在文件中。

d. 选择要将集群部署到的 AWS 区域。

e. 选择您为集群配置的 Route 53 服务的基域。

f. 为集群输入一个描述性名称。

g. 粘贴 Red Hat OpenShift Cluster Manager 中的 pull secret 。

注意

如果您在主机上配置的云供应商帐户没有足够的权限来部署集群，安装过程将会停止，并且显示缺少的权限。

集群部署完成后，终端会显示访问集群的信息，包括指向其 Web 控制台的链接和 **kubeadmin** 用户的凭证。

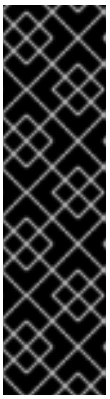
输出示例

```
...
INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run 'export
KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'
INFO Access the OpenShift web-console here: https://console-openshift-
console.apps.mycluster.example.com
```

```
INFO Login to the console with user: "kubeadmin", and password: "4vYBz-Ee6gm-ymBZj-Wt5AL"
INFO Time elapsed: 36m22s
```

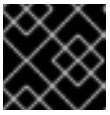
注意

当安装成功时，集群访问和凭证信息还会输出到 **<installation_directory>/.openshift_install.log**。



重要

- 安装程序生成的 Ignition 配置文件包含在 24 小时后过期的证书，然后在过期时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外情况是，您需要手动批准待处理的 **node-bootstrapper** 证书签名请求（CSR）来恢复 kubelet 证书。如需更多信息，请参阅*从过期的 control plane 证书中恢复*的文档。
- 建议您在生成 12 小时后使用 Ignition 配置文件，因为集群安装后 24 小时证书从 16 小时轮转至 22 小时。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中运行证书更新时避免安装失败。



重要

您不得删除安装程序或安装程序所创建的文件。需要这两者才能删除集群。

2. 可选：从您用来安装集群的 IAM 帐户删除或禁用 **AdministratorAccess** 策略。

注意

只有在安装过程中才需要 **AdministratorAccess** 策略提供的升级权限。

其他资源

- 如需有关 AWS 配置集和凭证配置的更多信息，请参阅 AWS 文档中的配置和凭证文件设置。

2.3.6. 通过下载二进制文件安装 OpenShift CLI

您需要安装 CLI（**oc**） 来使用命令行界面与 OpenShift Container Platform 进行交互。您可在 Linux、Windows 或 macOS 上安装 **oc**。

重要

如果安装了旧版本的 **oc**，则无法使用 OpenShift Container Platform 4.6 中的所有命令。下载并安装新版本的 **oc**。

2.3.6.1. 在 Linux 上安装 OpenShift CLI

您可以按照以下流程在 Linux 上安装 OpenShift CLI（**oc**）二进制文件。

流程

1. 进入到红帽客户门户网站上的 OpenShift Container Platform 下载页面。

2. 在 **Version** 下拉菜单中选择相应的版本。
3. 单击 **OpenShift v4.6 Linux 客户端** 条目旁边的 **Download Now**，再保存文件。
4. 解包存档：

```
$ tar xvzf <file>
```

5. 把 **oc** 二进制代码放到 **PATH** 中的目录中。
   执行以下命令可以查看当前的 **PATH** 设置：

```
$ echo $PATH
```

安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
$ oc <command>
```

2.3.6.2. 在 Windows 上安装 OpenShift CLI

您可以按照以下流程在 Windows 上安装 OpenShift CLI（**oc**）二进制代码。

流程

1. 进入到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 在 **Version** 下拉菜单中选择相应的版本。
3. 单击 **OpenShift v4.6 Windows 客户端**条目旁边的 **Download Now**，再保存文件。
4. 使用 ZIP 程序解压存档。
5. 把 **oc** 二进制代码放到 **PATH** 中的目录中。
   要查看您的 **PATH**，请打开命令提示窗口并执行以下命令：

```
C:\> path
```

安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
C:\> oc <command>
```

2.3.6.3. 在 macOS 上安装 OpenShift CLI

您可以按照以下流程在 macOS 上安装 OpenShift CLI（**oc**）二进制代码。

流程

1. 进入到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 在 **Version** 下拉菜单中选择相应的版本。
3. 单击 **OpenShift v4.6 MacOSX 客户端**条目旁边的 **Download Now**，再保存文件。
4. 解包和解压存档。

5. 将 **oc** 二进制文件移到 PATH 的目录中。
   要查看您的 **PATH**，打开一个终端窗口并执行以下命令：

```
$ echo $PATH
```

安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
$ oc <command>
```

2.3.7. 使用 CLI 登录到集群

您可以通过导出集群 **kubeconfig** 文件，以默认系统用户身份登录集群。**kubeconfig** 文件包含关于集群的信息，供 CLI 用于将客户端连接到正确集群和 API 服务器。该文件特只适用于一个特定的集群，在 OpenShift Container Platform 安装过程中创建。

先决条件

- 已部署了 OpenShift Container Platform 集群。
- 已安装 **oc** CLI。

流程

1. 导出 **kubeadmin** 凭证：

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

   1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

2. 使用导出的配置，验证能否成功运行 **oc** 命令：

```
$ oc whoami
```

   输出示例

```
system:admin
```

2.3.8. 使用 Web 控制台登录到集群

kubeadmin 用户默认在 OpenShift Container Platform 安装后存在。您可以使用 OpenShift Container Platform Web 控制台以 **kubeadmin** 用户身份登录集群。

先决条件

- 有访问安装主机的访问权限。
- 您完成了集群安装，所有集群 Operator 都可用。

流程

1. 从安装主机上的 **kubeadmin -password** 文件中获取 kubeadmin 用户的密码：

```
$ cat <installation_directory>/auth/kubeadmin-password
```

注意

另外，您还可以从安装主机上的 **<installation_directory>/.openshift_install.log** 日志文件获取 **kubeadmin** 密码。

2. 列出 OpenShift Container Platform Web 控制台路由：

```
$ oc get routes -n openshift-console | grep 'console-openshift'
```

注意

另外，您还可以从安装主机上的 **<installation_directory>/.openshift_install.log 日志** 文件获取 OpenShift Container Platform 路由。

输出示例

```
console     console-openshift-console.apps.<cluster_name>.<base_domain>           console
https   reencrypt/Redirect   None
```

3. 在 Web 浏览器中导航到上一命令输出中包括的路由，以 **kubeadmin** 用户身份登录。

其他资源

- 如需有关访问和了解 OpenShift Container Platform Web 控制台的更多信息，请参阅访问 Web 控制台。

2.3.9. OpenShift Container Platform 的 Telemetry 访问

在 OpenShift Container Platform 4.6 中，默认运行的 Telemetry 服务提供有关集群健康状况和成功更新的指标，需要访问互联网。如果您的集群连接到互联网，Telemetry 会自动运行，而且集群会注册到 OpenShift Cluster Manager。

确认 OpenShift Cluster Manager 清单正确后，可以由 Telemetry 自动维护，也可以使用 OpenShift Cluster Manager 手动维护，使用订阅监控来跟踪帐户或多集群级别的 OpenShift Container Platform 订阅。

其他资源

- 有关 Telemetry 服务的更多信息，请参阅关于远程健康监控。

2.3.10. 后续步骤

- 验证安装。
- 自定义集群。
- 如果需要，您可以选择不使用远程健康报告。
- 如果需要，您可以删除云供应商凭证。

2.4. 使用自定义在 AWS 上安装集群

在 OpenShift Container Platform 版本 4.6 中，您可以在安装程序在 Amazon Web Services（AWS）中置备的基础架构上安装自定义集群。要自定义安装，请在安装集群前修改 **install-config.yaml** 文件中的参数。

2.4.1. 先决条件

- 查看有关 OpenShift Container Platform 安装和更新流程的详细信息。
- 配置 AWS 帐户以托管集群。

重要

如果您的计算机上存储有 AWS 配置集，则不要在使用多因素验证设备的同时使用您生成的临时会话令牌。在集群的整个生命周期中，集群会持续使用您的当前 AWS 凭证来创建 AWS 资源，因此您必须使用长期凭证。要生成适当的密钥，请参阅 AWS 文档中的管理 IAM 用户的访问密钥。您可在运行安装程序时提供密钥。

- 如果使用防火墙，则必须将其配置为允许集群需要访问的站点。
- 如果不允许系统管理身份和访问管理（IAM），集群管理员可以 手动创建和维护 IAM 凭证。手动模式也可以用于云 IAM API 无法访问的环境中。

2.4.2. OpenShift Container Platform 的互联网访问

在 OpenShift Container Platform 4.6 中，您需要访问互联网来安装集群。

您必须具有以下互联网访问权限：

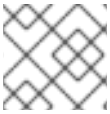
- 访问 OpenShift Cluster Manager 以下载安装程序并执行订阅管理。如果集群可以访问互联网，并且没有禁用 Telemetry，该服务会自动授权您的集群。
- 访问 Quay.io，以获取安装集群所需的软件包。
- 获取执行集群更新所需的软件包。

重要

如果您的集群无法直接访问互联网，则可以在置备的某些类基础架构上执行受限网络安装。在此过程中，您要下载所需的内容，并使用它在镜像 registry（mirror registry） 中填充安装集群并生成安装程序所需的软件包。对于某些安装类型，集群要安装到的环境不需要访问互联网。在更新集群之前，要更新 registry 镜像系统中的内容。

2.4.3. 生成 SSH 私钥并将其添加到代理中

如果要在集群上执行安装调试或灾难恢复，则必须为 **ssh-agent** 和安装程序提供 SSH 密钥。您可以使用此密钥访问公共集群中的 bootstrap 机器来排除安装问题。



注意

在生产环境中，您需要进行灾难恢复和调试。

您可以使用此密钥以 **core** 用户身份通过 SSH 连接到 master 节点。在部署集群时，此密钥会添加到 **core** 用户的 **~/.ssh/authorized_keys** 列表中。

注意

您必须使用一个本地密钥，而不要使用在特定平台上配置的密钥，如 AWS 密钥对。

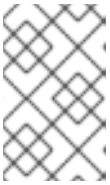
流程

1. 如果还没有为计算机上免密码身份验证而配置的 SSH 密钥，请创建一个。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ ssh-keygen -t ed25519 -N '' \
  -f <path>/<file_name> 1
```

1 指定新 SSH 密钥的路径和文件名，如 **~/.ssh/id_rsa**。如果您已有密钥对，请确保您的公钥位于 **~/.ssh** 目录中。

运行此命令会在指定的位置生成不需要密码的 SSH 密钥。



注意

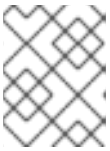
如果您计划在 **x86_64** 架构中安装使用 FIPS 验证的/Modules in Process 加密库的 OpenShift Container Platform 集群，不要创建使用 **ed25519** 算法的密钥。反之，创建一个使用 **rsa** 或 **ecdsa** 算法的密钥。

2. 作为后台任务启动 **ssh-agent** 进程：

```
$ eval "$(ssh-agent -s)"
```

输出示例

```
Agent pid 31874
```



注意

如果您的集群采用 FIPS 模式，则只使用 FIPS 兼容算法来生成 SSH 密钥。密钥必须是 RSA 或 ECDSA。

3. 将 SSH 私钥添加到 **ssh-agent**：

```
$ ssh-add <path>/<file_name> 1
```

输出示例

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

1 指定 SSH 私钥的路径和文件名，如 **~/.ssh/id_rsa**

后续步骤

- 在安装 OpenShift Container Platform 时，为安装程序提供 SSH 公钥。

2.4.4. 获取安装程序

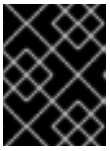
在安装 OpenShift Container Platform 之前，将安装文件下载到本地计算机上。

先决条件

- 运行 Linux 或 macOS 的计算机，本地磁盘空间为 500 MB

流程

1. 访问 OpenShift Cluster Manager 站点的 Infrastructure Provider 页面。如果您有红帽帐号，请使用自己的凭证登录。如果没有，请创建一个帐户。
2. 选择您的基础架构供应商。
3. 进入适用于您的安装类型的页面，下载您的操作系统的安装程序，并将文件放在要保存安装配置文件的目录中。。



重要

安装程序会在用来安装集群的计算机上创建若干文件。在完成集群安装后，您必须保留安装程序和安装程序所创建的文件。这两个文件都需要删除集群。

重要

删除安装程序创建的文件不会删除您的集群，即使集群在安装过程中失败也是如此。要删除集群，为特定云供应商完成 OpenShift Container Platform 卸载流程。

4. 提取安装程序。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ tar xvf openshift-install-linux.tar.gz
```

5. 从 Red Hat OpenShift Cluster Manager 下载安装 pull secret 。通过此 pull secret，您可以进行所含授权机构提供的服务的身份验证，这些服务包括为 OpenShift Container Platform 组件提供容器镜像的 Quay.io。

2.4.5. 创建安装配置文件

您可以自定义在 Amazon Web Services (AWS) 上安装的 OpenShift Container Platform 集群。

先决条件

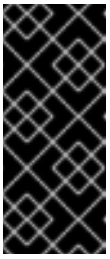
- 获取 OpenShift Container Platform 安装程序以及集群的 pull secret。

流程

1. 创建 **install-config.yaml** 文件。
   a. 更改到包含安装程序的目录，再运行以下命令：

```
$ ./openshift-install create install-config --dir <installation_directory> 1
```

1 对于 **<installation_directory>**，请指定用于保存安装程序所创建的文件的目录名称。

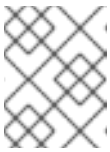


重要

指定一个空目录。一些安装信息，如 bootstrap X.509 证书，有较短的过期间隔，因此不要重复使用安装目录。如果要重复使用另一个集群安装中的个别文件，可以将其复制到您的目录中。但是，一些安装数据的文件名可能会在发行版本之间有所改变。从 OpenShift Container Platform 老版本中复制安装文件时要格外小心。

b. 在提示符处，提供您的云的配置详情：

i. 可选：选择用来访问集群机器的 SSH 密钥。



注意

对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 **ssh-agent** 进程使用的 SSH 密钥。

ii. 选择 **AWS** 作为目标平台。

iii. 如果计算机上没有保存 Amazon Web Services (AWS) 配置集，请为您配置用于运行安装程序的用户输入 AWS 访问密钥 ID 和 Secret 访问密钥。

iv. 选择要将集群部署到的 AWS 区域。

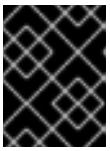
v. 选择您为集群配置的 Route 53 服务的基域。

vi. 为集群输入一个描述性名称。

vii. 粘贴 Red Hat OpenShift Cluster Manager 中的 pull secret 。

2. 修改 **install-config.yaml** 文件。您可以在**安装配置参数**部分中找到有关可用参数的更多信息。

3. 备份 **install-config.yaml** 文件，以便用于安装多个集群。



重要

install-config.yaml 文件会在安装过程中消耗掉。如果要重复使用此文件，必须现在备份。

2.4.5.1. 安装配置参数

在部署 OpenShift Container Platform 集群前，您可以提供参数值，以描述托管集群的云平台的帐户并选择性地自定义集群平台。在创建 **install-config.yaml** 安装配置文件时，您可以通过命令行来提供所需的参数的值。如果要自定义集群，可以修改 **install-config.yaml** 文件来提供关于平台的更多信息。

注意

安装之后，您无法修改 **install-config.yaml** 文件中的这些参数。

重要

openshift-install 命令不验证参数的字段名称。如果指定了不正确的名称，则不会创建相关的文件或对象，且不会报告错误。确保所有指定的参数的字段名称都正确。

2.4.5.1.1. 所需的配置参数

下表描述了所需的安装配置参数：

表 2.1. 所需的参数

参数	描述	值
apiVersion	**install-config.yaml** 内容的 API 版本。当前版本是 **v1**。安装程序还可能支持旧的 API 版本。	字符串
baseDomain	云供应商的基域。此基础域用于创建到 OpenShift Container Platform 集群组件的路由。集群的完整 DNS 名称是 **baseDomain** 和 **metadata.name** 参数值的组合，其格式为 **<metadata.name>.<baseDomain>**。	完全限定域名或子域名，如 **example.com**。
metadata	Kubernetes 资源 **ObjectMeta**，其中只消耗 **name** 参数。	对象
metadata.name	集群的名称。集群的 DNS 记录是 **{{.metadata.name}}.{{.baseDomain}}** 的子域。	小写字母,连字符(-)和句点(.)的字符串，如 **dev**。
platform	执行安装的具体平台配置：**aws**、**baremetal**、**azure**、**openstack**、**ovirt**、**vsphere**。有关 **platform.<platform>** 参数的额外信息，请参考下表来了解您的具体平台。	对象

参数	描述	值

pullSecret	从 Red Hat OpenShift Cluster Manager 获取 pull secret，验证从 Quay.io 等服务中下载 OpenShift Container Platform 组件的容器镜像。	<pre>{
		  "auths":{
		  "cloud.openshift.com":{
		  "auth":"b3Blb=",
		  "email":"you@example.com"
		  },
		  "quay.io":{
		  "auth":"b3Blb=",
		  "email":"you@example.com"
		  }
		  }
		}</pre>

2.4.5.1.2. 网络配置参数

您可以根据现有网络基础架构的要求自定义安装配置。例如，您可以扩展集群网络的 IP 地址块，或者提供不同于默认值的不同 IP 地址块。

只支持 IPv4 地址。

表 2.2. 网络参数

参数	描述	值
networking	集群网络的配置。	对象 **注意** 您不能在安装后修改 **networking** 对象指定的参数。
networking.networkType	要安装的集群网络供应商 Container Network Interface（CNI）插件。	**OpenShiftSDN** 或 **OVNKubernetes**。默认值为 **OpenShiftSDN**。

networking.clusterNetwork	pod 的 IP 地址块。
	
	默认值为 **10.128.0.0/14**，主机前缀为 **/23**。
	
	如果您指定多个 IP 地址块，则块不得互相重叠。
		一个对象数组。例如：
		
		<pre>networking:
		  clusterNetwork:
		  - cidr: 10.128.0.0/14
		  hostPrefix: 23</pre>

networking.clusterNetwork.cidr	使用 **networking.clusterNetwork** 时需要此项。IP 地址块。 一个 IPv4 网络。	使用 CIDR 形式的 IP 地址块。IPv4 块的前缀长度介于 **0** 到 **32** 之间。

参数	描述	值
networking.clusterNetwork.hostPrefix	分配给每个单独节点的子网前缀长度。例如，如果 **hostPrefix** 设为 **23**，则每个节点从所给的 **cidr** 中分配一个 **/23** 子网。**hostPrefix** 值 **23** 提供 510（2^(32 - 23)- 2）个 pod IP 地址。	子网前缀。 默认值为 **23**。
networking.serviceNetwork	服务的 IP 地址块。默认值为 **172.30.0.0/16**。 OpenShift SDN 和 OVN-Kubernetes 网络供应商只支持服务网络的一个 IP 地址块。	CIDR 格式具有 IP 地址块的数组。例如： networking: serviceNetwork: - 172.30.0.0/16
networking.machineNetwork	机器的 IP 地址块。 如果您指定多个 IP 地址块，则块不得互相重叠。	一个对象数组。例如： networking: machineNetwork: - cidr: 10.0.0.0/16
networking.machineNetwork.cidr	使用 **networking.machineNetwork** 时需要。IP 地址块。libvirt 以外的所有平台的默认值为 **10.0.0.0/16**。对于 libvirt，默认值为 **192.168.126.0/24**。	CIDR 表示法中的 IP 网络块。 例如： **10.0.0.0/16**。 **注意** 将 **networking.machineNetwork** 设置为与首选 NIC 所在的 CIDR 匹配。

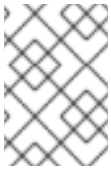
2.4.5.1.3. 可选配置参数

下表描述了可选安装配置参数：

表 2.3. 可选参数

参数	描述	值
additionalTrustBundle	添加到节点可信证书存储中的 PEM 编码 X.509 证书捆绑包。配置了代理时，也可以使用这个信任捆绑包。	字符串
compute	组成计算节点的机器的配置。	machine-pool 对象的数组。详情请查看以下"Machine-pool"表。

参数	描述	值
compute.architecture	决定池中机器的指令集合架构。目前不支持异构集群，因此所有池都必须指定相同的架构。有效值为 **amd64** （默认值）。	字符串
compute.hyperthreading	是否在计算机器上启用或禁用并发多线程或**超线程**。默认情况下，启用并发多线程以提高机器内核的性能。  **重要** 如果禁用并发多线程，请确保在容量规划时考虑到机器性能可能会显著降低的问题。	**Enabled** 或 **Disabled**
compute.name	使用 **compute** 时需要此值。机器池的名称。	**worker**
compute.platform	使用 **compute** 时需要此值。使用此参数指定托管 worker 机器的云供应商。此参数值必须与 **controlPlane.platform** 参数值匹配。	**aws**、**azure**、**gcp**、**openstack**、**ovirt**、**vsphere** 或 **{}**
compute.replicas	要置备的计算机器数量，也称为 worker 机器。	大于或等于 **2** 的正整数。默认值为 **3**。
controlPlane	组成 control plane 的机器的配置。	**MachinePool** 对象的数组。详情请查看以下"Machine-pool"表。
controlPlane.architecture	决定池中机器的指令集合架构。目前不支持异构集群，因此所有池都必须指定相同的架构。有效值为 **amd64** （默认值）。	字符串
controlPlane.hyperthreading	是否在 control plane 机器上启用或禁用并发多线程或**超线程**。默认情况下，启用并发多线程以提高机器内核的性能。  **重要** 如果禁用并发多线程，请确保在容量规划时考虑到机器性能可能会显著降低的问题。	**Enabled** 或 **Disabled**

参数	描述	值
controlPlane.name	使用 **controlPlane** 时需要。机器池的名称。	**master**
controlPlane.platform	使用 **controlPlane** 时需要。使用此参数指定托管 control plane 机器的云供应商。此参数值必须与 **compute.platform** 参数值匹配。	**aws**、**azure**、**gcp**、**openstack**、**ovirt**、**vsphere** 或 **{}**
controlPlane.replicas	要置备的 control plane 机器数量。	唯一支持的值是 **3**，它是默认值。
credentialsMode	Cloud Credential Operator（CCO）模式。如果没有指定任何模式，CCO 会动态地尝试决定提供的凭证的功能，在支持多个模式的平台上使用 mint 模式。  **注意** 不是所有 CCO 模式都支持所有云供应商。如需有关 CCO 模式的更多信息，请参阅*Red Hat Operator 参考指南*内容中的 *Cloud Credential Operator* 条目。	**Mint**、**Passthrough**、**Manual** 或空字符串(**""**)。
fips	启用或禁用 FIPS 模式。默认为 **false**（禁用）。如果启用了 FIPS 模式，运行 OpenShift Container Platform 的 Red Hat Enterprise Linux CoreOS (RHCOS) 机器会绕过默认的 Kubernetes 加密套件，并使用由 RHCOS 提供的加密模块。  **重要** 只有在 **x86_64** 架构中的 OpenShift Container Platform 部署支持 FIPS 验证的/Modules in Process 加密库。  **注意** 如果使用 Azure File 存储，则无法启用 FIPS 模式。	**false** 或 **true**

参数	描述	值
imageContentSources	release-image 内容的源和仓库。	对象数组。包括一个 **source** 以及可选的 **mirrors**，如下表所示。
imageContentSources.source	使用 **imageContentSources** 时需要。指定用户在镜像拉取规格中引用的仓库。	字符串
imageContentSources.mirrors	指定可能还包含同一镜像的一个或多个仓库。	字符串数组
publish	如何发布或公开集群的面向用户的端点，如 Kubernetes API、OpenShift 路由。	**Internal** 或 **External**。把 **publish** 设置为 **Internal** 以部署一个私有集群，它不能被互联网访问。默认值为 **External**。
sshKey	用于验证集群机器访问的 SSH 密钥或密钥。 **注意** 对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 **ssh-agent** 进程使用的 SSH 密钥。	一个或多个密钥。例如： sshKey: <key1> <key2> <key3>

2.4.5.1.4. 可选的 AWS 配置参数

下表描述了可选的 AWS 配置参数：

表 2.4. 可选的 AWS 参数

参数	描述	值
compute.platform.aws.amiID	用于为集群引导计算机器的 AWS AMI。对于需要自定义 RHCOS AMI 的区域来说，这是必需的。	属于集合 AWS 区域的任何已发布或自定义 RHCOS AMI。
compute.platform.aws.rootVolume.iops	为根卷保留的每秒输入/输出操作 (IOPS) 数。	整数，如 **4000**。

参数	描述	值
compute.platform.aws.rootVolume.size	以 GiB 为单位的根卷大小。	整数，如 **500**。
compute.platform.aws.rootVolume.type	根卷的类型。	有效的 AWS EBS 卷类型，如 **io1**。
compute.platform.aws.rootVolume.kmsKeyARN	KMS 密钥的 Amazon 资源名称（密钥 ARN）。这是使用特定 KMS 密钥加密 worker 节点的操作系统卷。	有效的 密钥 ID 或密钥 ARN。
compute.platform.aws.type	计算机器的 EC2 实例类型。	有效的 AWS 实例类型，如 **c5.9xlarge**。
compute.platform.aws.zones	安装程序在其中为计算机机器池创建机器的可用区。如果您提供自己的 VPC，则必须在那个可用域中提供一个子网。	有效 AWS 可用区的列表，如 **us-east-1c**，以 YAML 序列表示。
compute.aws.region	安装程序在其中创建计算资源的 AWS 区域。	任何有效的 AWS 区域，如 **us-east-1**。
controlPlane.platform.aws.amiID	用于为集群引导 control plane 机器的 AWS AMI。对于需要自定义 RHCOS AMI 的区域来说，这是必需的。	属于集合 AWS 区域的任何已发布或自定义 RHCOS AMI。
controlPlane.platform.aws.rootVolume.kmsKeyARN	KMS 密钥的 Amazon 资源名称（密钥 ARN）。这需要使用特定的 KMS 密钥加密 control plane 节点的操作系统卷。	有效的 密钥 ID 和密钥 ARN。
controlPlane.platform.aws.type	control plane 机器的 EC2 实例类型。	有效的 AWS 实例类型，如 **c5.9xlarge**。
controlPlane.platform.aws.zones	安装程序在其中为 control plane 机器池创建机器的可用区。	有效 AWS 可用区的列表，如 **us-east-1c**，以 YAML 序列表示。
controlPlane.aws.region	安装程序在其中创建 control plane 资源的 AWS 区域。	有效的 AWS 区域，如 **us-east-1**。

参数	描述	值
platform.aws.amiID	用于为集群引导所有机器的 AWS AMI。如果设置，AMI 必须属于与集群相同的区域。对于需要自定义 RHCOS AMI 的区域来说，这是必需的。	属于集合 AWS 区域的任何已发布或自定义 RHCOS AMI。
platform.aws.serviceEndpoints.name	AWS 服务端点名称。只有在必须使用替代 AWS 端点（如 FIPS）时，才需要自定义端点。可以为 EC2、S3、IAM、Elastic Load Balancing、Tagging、Route 53 和 STS AWS 服务指定自定义 API 端点。	有效的 AWS 服务端点名称。
platform.aws.serviceEndpoints.url	AWS 服务端点 URL。URL 必须使用 **https** 协议，主机必须信任该证书。	有效的 AWS 服务端点 URL。
platform.aws.userTags	键与值的映射，安装程序将其作为标签添加到它所创建的所有资源。	任何有效的 YAML 映射，如 **<key>: <value>** 格式的键值对。如需有关 AWS 标签的更多信息，请参阅 AWS 文档中的标记您的 Amazon EC2 资源。
platform.aws.subnets	如果您提供 VPC，而不是让安装程序为您创建 VPC，请指定要使用的集群子网。子网必须是您指定的同一 **machineNetwork[].cidr** 范围的一部分。对于标准集群，为每个可用区指定一个公共和私有子网。对于私有集群，为每个可用区指定一个私有子网。	有效的子网 ID。

2.4.5.2. AWS 的自定义 install-config.yaml 文件示例

您可以自定义 **install-config.yaml** 文件，以指定有关 OpenShift Container Platform 集群平台的更多信息，或修改所需参数的值。

重要

此示例 YAML 文件仅供参考。您必须使用安装程序来获取 **install-config.yaml** 文件，并且修改该文件。

```
apiVersion: v1
baseDomain: example.com (1)
credentialsMode: Mint (2)
controlPlane: (3) (4)
  hyperthreading: Enabled (5)
```

```
  name: master
  platform:
    aws:
      zones:
      - us-west-2a
      - us-west-2b
      rootVolume:
        iops: 4000
        size: 500
        type: io1 6
      type: m5.xlarge
  replicas: 3
compute: 7
- hyperthreading: Enabled 8
  name: worker
  platform:
    aws:
      rootVolume:
        iops: 2000
        size: 500
        type: io1 9
      type: c5.4xlarge
      zones:
      - us-west-2c
  replicas: 3
metadata:
  name: test-cluster 10
networking:
  clusterNetwork:
  - cidr: 10.128.0.0/14
    hostPrefix: 23
  machineNetwork:
  - cidr: 10.0.0.0/16
  networkType: OpenShiftSDN
  serviceNetwork:
  - 172.30.0.0/16
platform:
  aws:
    region: us-west-2 11
    userTags:
      adminContact: jdoe
      costCenter: 7536
    amiID: ami-96c6f8f7 12
    serviceEndpoints: 13
      - name: ec2
        url: https://vpce-id.ec2.us-west-2.vpce.amazonaws.com
fips: false 14
sshKey: ssh-ed25519 AAAA... 15
pullSecret: '{"auths": ...}' 16
```

1 10 11 16 必需。安装程序会提示您输入这个值。

2 可选：添加此参数来强制 Cloud Credential Operator（CCO）使用指定的模式，而不是让 CCO 动态尝试决定凭证的功能。如需有关 CCO 模式的详情，请参阅 *Red Hat Operator* 参考内容中的 *Cloud Credential Operator* 条目。

3 7 如果没有提供这些参数和值，安装程序会提供默认值。

4 **controlPlane** 部分是一个单映射，但 **compute** 部分是一系列映射。为满足不同数据结构的要求，**compute** 部分的第一行必须以连字符 **-** 开头，**controlPlane** 部分的第一行则不可以连字符开头。只使用一个 control plane 池。

5 8 是否要启用或禁用并发多线程或**超线程**。默认情况下，启用并发多线程以提高机器内核的性能。您可以通过将参数值设为 **Disabled** 来禁用。如果您在某些集群机器上禁用并发多线程，则必须在所有集群机器上禁用。

重要

如果禁用并发多线程，请确保在容量规划时考虑到机器性能可能会显著降低的问题。如果您对机器禁用并发多线程，请使用较大的实例类型，如 **m4.2xlarge** 或 **m5.2xlarge**。

6 9 要为 etcd 配置更快的存储，特别是对于较大的集群，请将存储类型设置为 **io1**，并将 **iops** 设为 **2000**。

12 用于为集群引导机器的 AMI ID。如果设置，AMI 必须属于与集群相同的区域。

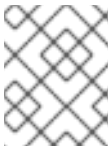
13 AWS 服务端点。在安装到未知 AWS 区域时，需要自定义端点。端点 URL 必须使用 **https** 协议，主机必须信任该证书。

14 是否启用或禁用 FIPS 模式。默认情况下不启用 FIPS 模式。如果启用了 FIPS 模式，运行 OpenShift Container Platform 的 Red Hat Enterprise Linux CoreOS (RHCOS) 机器会绕过默认的 Kubernetes 加密套件，并使用由 RHCOS 提供的加密模块。

重要

只有在 **x86_64** 架构中的 OpenShift Container Platform 部署支持 FIPS 验证的/Modules in Process 加密库。

15 您可以选择提供您用来访问集群中机器的 **sshKey** 值。



注意

对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 **ssh-agent** 进程使用的 SSH 密钥。

2.4.5.3. 在安装过程中配置集群范围代理

生产环境可能会拒绝直接访问互联网，而是提供 HTTP 或 HTTPS 代理。您可以通过在 **install-config.yaml** 文件中配置代理设置，将新的 OpenShift Container Platform 集群配置为使用代理。

先决条件

- 您有一个现有的 **install-config.yaml** 文件。
- 您检查了集群需要访问的站点，并决定是否需要绕过代理。默认情况下代理所有集群出口流量，包括对托管云供应商 API 的调用。您需要将站点添加到 **Proxy** 对象的 **spec.noProxy** 字段来绕过代理。

注意

Proxy 对象 **status.noProxy** 字段使用安装配置中的 **networking.machineNetwork[].cidr**、**networking.clusterNetwork[].cidr** 和 **networking.serviceNetwork[]** 字段的值填充。

对于在 Amazon Web Services(AWS)、Google Cloud Platform(GCP)、Microsoft Azure 和 Red Hat OpenStack Platform(RHOSP)上安装, **Proxy** 对象 **status.noProxy** 字段也会使用实例元数据端点填充(**169.254.169.254**)。

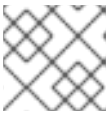
- 如果您的集群位于 AWS 上，请将 **ec2.<region>.amazonaws.com**、**elasticloadbalancing.<region>.amazonaws.com** 和 **s3.<region>.amazonaws.com** 端点添加到 VPC 端点。需要这些端点才能完成节点到 AWS EC2 API 的请求。由于代理在容器级别而不是节点级别工作，因此您必须通过 AWS 专用网络将这些请求路由到 AWS EC2 API。在代理服务器中的允许列表中添加 EC2 API 的公共 IP 地址是不够的。

流程

1. 编辑 **install-config.yaml** 文件并添加代理设置。例如：

```
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: example.com 3
additionalTrustBundle: | 4
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
...
```

1. 用于创建集群外 HTTP 连接的代理 URL。URL 必须是 **http**。
2. 用于创建集群外 HTTPS 连接的代理 URL。
3. 要排除在代理中的目标域名、IP 地址或其他网络 CIDR 的逗号分隔列表。在域前面加 **.** 来仅匹配子域。例如： **.y.com** 匹配 **x.y.com**，但不匹配 **y.com**。使用 ***** 绕过所有目的地的代理。
4. 如果提供，安装程序会在 **openshift-config 命名空间中生成名为 user-ca- bundle** 的配置映射来保存额外的 CA 证书。如果您提供 **additionalTrustBundle** 和至少一个代理设置，则 **Proxy** 对象会被配置为引用 **trustedCA** 字段中的 **user-ca-bundle** 配置映射。然后，Cluster Network Operator 会创建一个 **trusted-ca-bundle** 配置映射，该配置映射将为 **trustedCA** 参数指定的内容与 RHCOS 信任捆绑包合并。 **additionalTrustBundle** 字段是必需的，除非代理的身份证书由来自 RHCOS 信任捆绑包的颁发机构签名。



注意

安装程序不支持代理的 **readinessEndpoints** 字段。

2. 保存该文件，并在安装 OpenShift Container Platform 时引用。

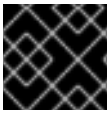
安装程序会创建一个名为 **cluster** 的集群范围代理，该代理使用提供的 **install-config.yaml** 文件中的代理设置。如果没有提供代理设置，仍然会创建一个 **cluster Proxy** 对象，但它会有一个空 **spec**。

注意

只支持名为 **cluster** 的 **Proxy** 对象，且无法创建额外的代理。

2.4.6. 部署集群

您可以在兼容云平台中安装 OpenShift Container Platform。



重要

安装程序的 **create cluster** 命令只能在初始安装过程中运行一次。

先决条件

- 配置托管集群的云平台的帐户。
- 获取 OpenShift Container Platform 安装程序以及集群的 pull secret。

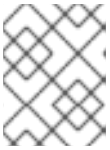
流程

1. 更改为包含安装程序的目录并初始化集群部署：

```
$ ./openshift-install create cluster --dir <installation_directory> \ 1
    --log-level=info 2
```

1 对于 **<installation_directory>**，请指定自定义 **./install-config.yaml** 文件的位置。

2 要查看不同的安装详情，请指定 **warn**、**debug** 或 **error**，而不要指定 **info**。



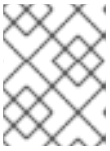
注意

如果您在主机上配置的云供应商帐户没有足够的权限来部署集群，安装过程将会停止，并且显示缺少的权限。

集群部署完成后，终端会显示访问集群的信息，包括指向其 Web 控制台的链接和 **kubeadmin** 用户的凭证。

输出示例

```
...
INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run 'export
KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'
INFO Access the OpenShift web-console here: https://console-openshift-
console.apps.mycluster.example.com
INFO Login to the console with user: "kubeadmin", and password: "4vYBz-Ee6gm-ymBZj-
Wt5AL"
INFO Time elapsed: 36m22s
```



注意

当安装成功时，集群访问和凭证信息还会输出到 **<installation_directory>/.openshift_install.log**。

重要

- 安装程序生成的 Ignition 配置文件包含在 24 小时后过期的证书，然后在过期时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外情况是，您需要手动批准待处理的 **node-bootstrapper** 证书签名请求（CSR）来恢复 kubelet 证书。如需更多信息，请参阅*从过期的 control plane 证书中恢复*的文档。
- 建议您在生成 12 小时后使用 Ignition 配置文件，因为集群安装后 24 小时证书从 16 小时轮转至 22 小时。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中运行证书更新时避免安装失败。

重要

您不得删除安装程序或安装程序所创建的文件。需要这两者才能删除集群。

2. 可选：从您用来安装集群的 IAM 帐户删除或禁用 **AdministratorAccess** 策略。

注意

只有在安装过程中才需要 **AdministratorAccess** 策略提供的升级权限。

2.4.7. 通过下载二进制文件安装 OpenShift CLI

您需要安装 CLI（**oc**） 来使用命令行界面与 OpenShift Container Platform 进行交互。您可在 Linux、Windows 或 macOS 上安装 **oc**。

重要

如果安装了旧版本的 **oc**，则无法使用 OpenShift Container Platform 4.6 中的所有命令。下载并安装新版本的 **oc**。

2.4.7.1. 在 Linux 上安装 OpenShift CLI

您可以按照以下流程在 Linux 上安装 OpenShift CLI（**oc**）二进制文件。

流程

1. 进入到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 在 **Version** 下拉菜单中选择相应的版本。
3. 单击 **OpenShift v4.6 Linux 客户端** 条目旁边的 **Download Now**，再保存文件。
4. 解包存档：

```
$ tar xvzf <file>
```

5. 把 **oc** 二进制代码放到 **PATH** 中的目录中。
   执行以下命令可以查看当前的 **PATH** 设置：

```
$ echo $PATH
```

安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
$ oc <command>
```

2.4.7.2. 在 Windows 上安装 OpenShift CLI

您可以按照以下流程在 Windows 上安装 OpenShift CLI（**oc**）二进制代码。

流程

1. 进入到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 在 **Version** 下拉菜单中选择相应的版本。
3. 单击 **OpenShift v4.6 Windows 客户端**条目旁边的 **Download Now**，再保存文件。
4. 使用 ZIP 程序解压存档。
5. 把 **oc** 二进制代码放到 **PATH** 中的目录中。
   要查看您的 **PATH**，请打开命令提示窗口并执行以下命令：

```
C:\> path
```

安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
C:\> oc <command>
```

2.4.7.3. 在 macOS 上安装 OpenShift CLI

您可以按照以下流程在 macOS 上安装 OpenShift CLI（**oc**）二进制代码。

流程

1. 进入到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 在 **Version** 下拉菜单中选择相应的版本。
3. 单击 **OpenShift v4.6 MacOSX 客户端**条目旁边的 **Download Now**，再保存文件。
4. 解包和解压存档。
5. 将 **oc** 二进制文件移到 PATH 的目录中。
   要查看您的 **PATH**，打开一个终端窗口并执行以下命令：

```
$ echo $PATH
```

安装 OpenShift CLI 后，可以使用 **oc** 命令：

-

```
$ oc <command>
```

2.4.8. 使用 CLI 登录到集群

您可以通过导出集群 **kubeconfig** 文件，以默认系统用户身份登录集群。**kubeconfig** 文件包含关于集群的信息，供 CLI 用于将客户端连接到正确集群和 API 服务器。该文件特只适用于一个特定的集群，在 OpenShift Container Platform 安装过程中创建。

先决条件

- 已部署了 OpenShift Container Platform 集群。
- 已安装 **oc** CLI。

流程

1. 导出 **kubeadmin** 凭证：

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

2. 使用导出的配置，验证能否成功运行 **oc** 命令：

```
$ oc whoami
```

输出示例

```
system:admin
```

2.4.9. 使用 Web 控制台登录到集群

kubeadmin 用户默认在 OpenShift Container Platform 安装后存在。您可以使用 OpenShift Container Platform Web 控制台以 **kubeadmin** 用户身份登录集群。

先决条件

- 有访问安装主机的访问权限。
- 您完成了集群安装，所有集群 Operator 都可用。

流程

1. 从安装主机上的 **kubeadmin -password** 文件中获取 kubeadmin 用户的密码：

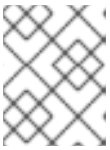
```
$ cat <installation_directory>/auth/kubeadmin-password
```

注意

另外，您还可以从安装主机上的 **<installation_directory>/.openshift_install.log** 日志文件获取 **kubeadmin** 密码。

2. 列出 OpenShift Container Platform Web 控制台路由：

```
$ oc get routes -n openshift-console | grep 'console-openshift'
```



注意

另外，您还可以从安装主机上的 **<installation_directory>/.openshift_install.log 日志** 文件获取 OpenShift Container Platform 路由。

输出示例

```
console     console-openshift-console.apps.<cluster_name>.<base_domain>          console
https   reencrypt/Redirect   None
```

3. 在 Web 浏览器中导航到上一命令输出中包括的路由，以 **kubeadmin** 用户身份登录。

其他资源

- 如需有关访问和了解 OpenShift Container Platform Web 控制台的更多信息，请参阅访问 Web 控制台。

2.4.10. OpenShift Container Platform 的 Telemetry 访问

在 OpenShift Container Platform 4.6 中，默认运行的 Telemetry 服务提供有关集群健康状况和成功更新的指标，需要访问互联网。如果您的集群连接到互联网，Telemetry 会自动运行，而且集群会注册到 OpenShift Cluster Manager。

确认 OpenShift Cluster Manager 清单正确后，可以由 Telemetry 自动维护，也可以使用 OpenShift Cluster Manager 手动维护，使用订阅监控来跟踪帐户或多集群级别的 OpenShift Container Platform 订阅。

其他资源

- 有关 Telemetry 服务的更多信息，请参阅关于远程健康监控。

2.4.11. 后续步骤

- 验证安装。
- 自定义集群。
- 如果需要，您可以选择不使用远程健康报告。
- 如果需要，您可以删除云供应商凭证。

2.5. 使用自定义网络在 AWS 上安装集群

在 OpenShift Container Platform 版本 4.6 中，您可以使用自定义网络配置选项在 Amazon Web Services（AWS）上安装集群。通过自定义网络配置，您的集群可以与环境中现有的 IP 地址分配共存，并与现有的 MTU 和 VXLAN 配置集成。

大部分网络配置参数必须在安装过程中设置，只有 **kubeProxy** 配置参数可以在运行的集群中修改。

2.5.1. 先决条件

- 查看有关 OpenShift Container Platform 安装和更新流程的详细信息。
- 配置 AWS 帐户以托管集群。

重要

如果您的计算机上存储有 AWS 配置集，则不要在使用多因素验证设备的同时使用您生成的临时会话令牌。在集群的整个生命周期中，集群会持续使用您的当前 AWS 凭证来创建 AWS 资源，因此您必须使用基于密钥的长期凭证。要生成适当的密钥，请参阅 AWS 文档中的管理 IAM 用户的访问密钥。您可在运行安装程序时提供密钥。

- 如果使用防火墙，则必须将其配置为允许集群需要访问的站点。
- 如果不允许系统管理身份和访问管理（IAM），集群管理员可以 手动创建和维护 IAM 凭证。手动模式也可以用于云 IAM API 无法访问的环境中。

2.5.2. OpenShift Container Platform 的互联网访问

在 OpenShift Container Platform 4.6 中，您需要访问互联网来安装集群。

您必须具有以下互联网访问权限：

- 访问 OpenShift Cluster Manager 以下载安装程序并执行订阅管理。如果集群可以访问互联网，并且没有禁用 Telemetry，该服务会自动授权您的集群。
- 访问 Quay.io，以获取安装集群所需的软件包。
- 获取执行集群更新所需的软件包。

重要

如果您的集群无法直接访问互联网，则可以在置备的某些类基础架构上执行受限网络安装。在此过程中，您要下载所需的内容，并使用它在镜像 registry（mirror registry） 中填充安装集群并生成安装程序所需的软件包。对于某些安装类型，集群要安装到的环境不需要访问互联网。在更新集群之前，要更新 registry 镜像系统中的内容。

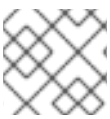
2.5.3. 生成 SSH 私钥并将其添加到代理中

如果要在集群上执行安装调试或灾难恢复，则必须为 **ssh-agent** 和安装程序提供 SSH 密钥。您可以使用此密钥访问公共集群中的 bootstrap 机器来排除安装问题。

注意

在生产环境中，您需要进行灾难恢复和调试。

您可以使用此密钥以 **core** 用户身份通过 SSH 连接到 master 节点。在部署集群时，此密钥会添加到 **core** 用户的 **~/.ssh/authorized_keys** 列表中。



注意

您必须使用一个本地密钥，而不要使用在特定平台上配置的密钥，如 AWS 密钥对。

流程

1. 如果还没有为计算机上免密码身份验证而配置的 SSH 密钥，请创建一个。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ ssh-keygen -t ed25519 -N '' \
  -f <path>/<file_name> 1
```

1 指定新 SSH 密钥的路径和文件名，如 **~/.ssh/id_rsa**。如果您已有密钥对，请确保您的公钥位于 **~/.ssh** 目录中。

运行此命令会在指定的位置生成不需要密码的 SSH 密钥。

注意

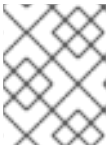
如果您计划在 **x86_64** 架构中安装使用 FIPS 验证的/Modules in Process 加密库的 OpenShift Container Platform 集群，不要创建使用 **ed25519** 算法的密钥。反之，创建一个使用 **rsa** 或 **ecdsa** 算法的密钥。

2. 作为后台任务启动 **ssh-agent** 进程：

```
$ eval "$(ssh-agent -s)"
```

输出示例

```
Agent pid 31874
```



注意

如果您的集群采用 FIPS 模式，则只使用 FIPS 兼容算法来生成 SSH 密钥。密钥必须是 RSA 或 ECDSA。

3. 将 SSH 私钥添加到 **ssh-agent**：

```
$ ssh-add <path>/<file_name> 1
```

输出示例

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

1 指定 SSH 私钥的路径和文件名，如 **~/.ssh/id_rsa**

后续步骤

- 在安装 OpenShift Container Platform 时，为安装程序提供 SSH 公钥。

2.5.4. 获取安装程序

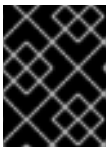
在安装 OpenShift Container Platform 之前，将安装文件下载到本地计算机上。

先决条件

- 运行 Linux 或 macOS 的计算机，本地磁盘空间为 500 MB

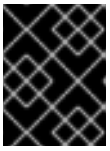
流程

1. 访问 OpenShift Cluster Manager 站点的 Infrastructure Provider 页面。如果您有红帽帐号，请使用自己的凭证登录。如果没有，请创建一个帐户。
2. 选择您的基础架构供应商。
3. 进入适用于您的安装类型的页面，下载您的操作系统的安装程序，并将文件放在要保存安装配置文件的目录中。。



重要

安装程序会在用来安装集群的计算机上创建若干文件。在完成集群安装后，您必须保留安装程序和安装程序所创建的文件。这两个文件都需要删除集群。



重要

删除安装程序创建的文件不会删除您的集群，即使集群在安装过程中失败也是如此。要删除集群，为特定云供应商完成 OpenShift Container Platform 卸载流程。

4. 提取安装程序。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ tar xvf openshift-install-linux.tar.gz
```

5. 从 Red Hat OpenShift Cluster Manager 下载安装 pull secret 。通过此 pull secret，您可以进行所含授权机构提供的服务的身份验证，这些服务包括为 OpenShift Container Platform 组件提供容器镜像的 Quay.io。

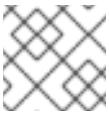
2.5.5. 网络配置阶段

当在安装前指定集群配置时，在安装过程中的几个阶段可以修改网络配置：

阶段 1

输入 **openshift-install create install-config** 命令后。在 **install-config.yaml** 文件中，您可以自定义以下与网络相关的字段：

- **networking.networkType**
- **networking.clusterNetwork**
- **networking.serviceNetwork**
- **networking.machineNetwork**
  有关这些字段的更多信息，请参阅"安装配置参数"。



注意

将 **networking.machineNetwork** 设置为与首选 NIC 所在的 CIDR 匹配。

阶段 2

输入 **openshift-install create manifests** 命令后。如果必须指定高级网络配置，在这个阶段中，只能使用您要修改的字段来定义自定义的 Cluster Network Operator 清单。

在 2 阶段，您无法覆盖 **install-config.yaml** 文件中的 1 阶段中指定的值。但是，您可以在第 2 阶段进一步自定义集群网络供应商。

2.5.6. 创建安装配置文件

您可以自定义在 Amazon Web Services (AWS) 上安装的 OpenShift Container Platform 集群。

先决条件

- 获取 OpenShift Container Platform 安装程序以及集群的 pull secret。

流程

1. 创建 **install-config.yaml** 文件。
   a. 更改到包含安装程序的目录，再运行以下命令：

```
$ ./openshift-install create install-config --dir <installation_directory> 1
```

   1 对于 **<installation_directory>**，请指定用于保存安装程序所创建的文件的目录名称。

重要

指定一个空目录。一些安装信息，如 bootstrap X.509 证书，有较短的过期间隔，因此不要重复使用安装目录。如果要重复使用另一个集群安装中的个别文件，可以将其复制到您的目录中。但是，一些安装数据的文件名可能会在发行版本之间有所改变。从 OpenShift Container Platform 老版本中复制安装文件时要格外小心。

   b. 在提示符处，提供您的云的配置详情：
      i. 可选：选择用来访问集群机器的 SSH 密钥。

注意

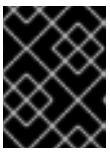
对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 **ssh-agent** 进程使用的 SSH 密钥。

      ii. 选择 **AWS** 作为目标平台。
      iii. 如果计算机上没有保存 Amazon Web Services (AWS) 配置集，请为您配置用于运行安装程序的用户输入 AWS 访问密钥 ID 和 Secret 访问密钥。
      iv. 选择要将集群部署到的 AWS 区域。
      v. 选择您为集群配置的 Route 53 服务的基域。
      vi. 为集群输入一个描述性名称。

vii. 粘贴 Red Hat OpenShift Cluster Manager 中的 pull secret 。

2. 修改 **install-config.yaml** 文件。您可以在**安装配置参数**部分中找到有关可用参数的更多信息。

3. 备份 **install-config.yaml** 文件，以便用于安装多个集群。



重要

install-config.yaml 文件会在安装过程中消耗掉。如果要重复使用此文件，必须现在备份。

2.5.6.1. 安装配置参数

在部署 OpenShift Container Platform 集群前，您可以提供参数值，以描述托管集群的云平台的帐户并选择性地自定义集群平台。在创建 **install-config.yaml** 安装配置文件时，您可以通过命令行来提供所需的参数的值。如果要自定义集群，可以修改 **install-config.yaml** 文件来提供关于平台的更多信息。

注意

安装之后，您无法修改 **install-config.yaml** 文件中的这些参数。

重要

openshift-install 命令不验证参数的字段名称。如果指定了不正确的名称，则不会创建相关的文件或对象，且不会报告错误。确保所有指定的参数的字段名称都正确。

2.5.6.1.1. 所需的配置参数

下表描述了所需的安装配置参数：

表 2.5. 所需的参数

参数	描述	值
apiVersion	**install-config.yaml** 内容的 API 版本。当前版本是 **v1**。安装程序还可能支持旧的 API 版本。	字符串
baseDomain	云供应商的基域。此基础域用于创建到 OpenShift Container Platform 集群组件的路由。集群的完整 DNS 名称是 **baseDomain** 和 **metadata.name** 参数值的组合，其格式为 **<metadata.name>.<baseDomain>**。	完全限定域名或子域名，如 **example.com**。
metadata	Kubernetes 资源 **ObjectMeta**，其中只消耗 **name** 参数。	对象

参数	描述	值
metadata.name	集群的名称。集群的 DNS 记录是 **{{.metadata.name}}.{{.baseDomain}}** 的子域。	小写字母,连字符(-)和句点(.)的字符串，如 **dev**。
platform	执行安装的具体平台配置：**aws**、**baremetal**、**azure**、**openstack**、**ovirt**、**vsphere**。有关 **platform.<platform>** 参数的额外信息，请参考下表来了解您的具体平台。	对象

pullSecret	从 Red Hat OpenShift Cluster Manager 获取 pull secret，验证从 Quay.io 等服务中下载 OpenShift Container Platform 组件的容器镜像。	<pre>{
		  "auths":{
		  "cloud.openshift.com":{
		  "auth":"b3Blb=",
		  "email":"you@example.com"
		  },
		  "quay.io":{
		  "auth":"b3Blb=",
		  "email":"you@example.com"
		  }
		  }
		}</pre>

2.5.6.1.2. 网络配置参数

您可以根据现有网络基础架构的要求自定义安装配置。例如，您可以扩展集群网络的 IP 地址块，或者提供不同于默认值的不同 IP 地址块。

只支持 IPv4 地址。

表 2.6. 网络参数

参数	描述	值
networking	集群网络的配置。	对象 **注意** 您不能在安装后修改 **networking** 对象指定的参数。
networking.networkType	要安装的集群网络供应商 Container Network Interface（CNI）插件。	**OpenShiftSDN** 或 **OVNKubernetes**。默认值为 **OpenShiftSDN**。

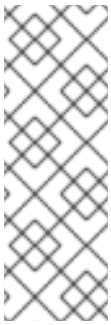
参数	描述	值
networking.clusterNetwork	pod 的 IP 地址块。 默认值为 **10.128.0.0/14**，主机前缀为 **/23**。 如果您指定多个 IP 地址块，则块不得互相重叠。	一个对象数组。例如： networking: clusterNetwork: - cidr: 10.128.0.0/14 hostPrefix: 23
networking.clusterNetwork.cidr	使用 **networking.clusterNetwork** 时需要此项。IP 地址块。 一个 IPv4 网络。	使用 CIDR 形式的 IP 地址块。IPv4 块的前缀长度介于 **0** 到 **32** 之间。
networking.clusterNetwork.hostPrefix	分配给每个单独节点的子网前缀长度。例如，如果 **hostPrefix** 设为 **23**，则每个节点从所给的 **cidr** 中分配一个 **/23** 子网。**hostPrefix** 值 **23** 提供 510（2^(32 - 23)- 2）个 pod IP 地址。	子网前缀。 默认值为 **23**。
networking.serviceNetwork	服务的 IP 地址块。默认值为 **172.30.0.0/16**。 OpenShift SDN 和 OVN-Kubernetes 网络供应商只支持服务网络的一个 IP 地址块。	CIDR 格式具有 IP 地址块的数组。例如： networking: serviceNetwork: - 172.30.0.0/16
networking.machineNetwork	机器的 IP 地址块。 如果您指定多个 IP 地址块，则块不得互相重叠。	一个对象数组。例如： networking: machineNetwork: - cidr: 10.0.0.0/16
networking.machineNetwork.cidr	使用 **networking.machineNetwork** 时需要。IP 地址块。libvirt 以外的所有平台的默认值为 **10.0.0.0/16**。对于 libvirt，默认值为 **192.168.126.0/24**。	CIDR 表示法中的 IP 网络块。 例如：**10.0.0.0/16**。 **注意** 将 **networking.machineNetwork** 设置为与首选 NIC 所在的 CIDR 匹配。

2.5.6.1.3. 可选配置参数

下表描述了可选安装配置参数：

表 2.7. 可选参数

参数	描述	值
additionalTrustBundle	添加到节点可信证书存储中的 PEM 编码 X.509 证书捆绑包。配置了代理时，也可以使用这个信任捆绑包。	字符串
compute	组成计算节点的机器的配置。	machine-pool 对象的数组。详情请查看以下"Machine-pool"表。
compute.architecture	决定池中机器的指令集合架构。目前不支持异构集群，因此所有池都必须指定相同的架构。有效值为 **amd64** （默认值）。	字符串
compute.hyperthreading	是否在计算机器上启用或禁用并发多线程或**超线程**。默认情况下，启用并发多线程以提高机器内核的性能。  **重要** 如果禁用并发多线程，请确保在容量规划时考虑到机器性能可能会显著降低的问题。	**Enabled** 或 **Disabled**
compute.name	使用 **compute** 时需要此值。机器池的名称。	**worker**
compute.platform	使用 **compute** 时需要此值。使用此参数指定托管 worker 机器的云供应商。此参数值必须与 **controlPlane.platform** 参数值匹配。	**aws**、**azure**、**gcp**、**openstack**、**ovirt**、**vsphere** 或 **{}**
compute.replicas	要置备的计算机器数量，也称为 worker 机器。	大于或等于 **2** 的正整数。默认值为 **3**。
controlPlane	组成 control plane 的机器的配置。	**MachinePool** 对象的数组。详情请查看以下"Machine-pool"表。
controlPlane.architecture	决定池中机器的指令集合架构。目前不支持异构集群，因此所有池都必须指定相同的架构。有效值为 **amd64** （默认值）。	字符串

参数	描述	值
controlPlane.hyperthreading	是否在 control plane 机器上启用或禁用并发多线程或**超线程**。默认情况下，启用并发多线程以提高机器内核的性能。  **重要** 如果禁用并发多线程，请确保在容量规划时考虑到机器性能可能会显著降低的问题。	**Enabled** 或 **Disabled**
controlPlane.name	使用 **controlPlane** 时需要。机器池的名称。	**master**
controlPlane.platform	使用 **controlPlane** 时需要。使用此参数指定托管 control plane 机器的云供应商。此参数值必须与 **compute.platform** 参数值匹配。	**aws**、**azure**、**gcp**、**openstack**、**ovirt**、**vsphere** 或 **{}**
controlPlane.replicas	要置备的 control plane 机器数量。	唯一支持的值是 **3**，它是默认值。
credentialsMode	Cloud Credential Operator（CCO）模式。如果没有指定任何模式，CCO 会动态地尝试决定提供的凭证的功能，在支持多个模式的平台上使用 mint 模式。  **注意** 不是所有 CCO 模式都支持所有云供应商。如需有关 CCO 模式的更多信息，请参阅*Red Hat Operator 参考指南*内容中的 *Cloud Credential Operator* 条目。	**Mint**、**Passthrough**、**Manual** 或空字符串(**""**)。

参数	描述	值
fips	启用或禁用 FIPS 模式。默认为 **false**（禁用）。如果启用了 FIPS 模式，运行 OpenShift Container Platform 的 Red Hat Enterprise Linux CoreOS (RHCOS) 机器会绕过默认的 Kubernetes 加密套件，并使用由 RHCOS 提供的加密模块。 **重要** 只有在 **x86_64** 架构中的 OpenShift Container Platform 部署支持 FIPS 验证的/Modules in Process 加密库。 **注意** 如果使用 Azure File 存储，则无法启用 FIPS 模式。	**false** 或 **true**
imageContentSources	release-image 内容的源和仓库。	对象数组。包括一个 **source** 以及可选的 **mirrors**，如下表所示。
imageContentSources.source	使用 **imageContentSources** 时需要。指定用户在镜像拉取规格中引用的仓库。	字符串
imageContentSources.mirrors	指定可能还包含同一镜像的一个或多个仓库。	字符串数组
publish	如何发布或公开集群的面向用户的端点，如 Kubernetes API、OpenShift 路由。	**Internal** 或 **External**。把 **publish** 设置为 **Internal** 以部署一个私有集群，它不能被互联网访问。默认值为 **External**。
sshKey	用于验证集群机器访问的 SSH 密钥或密钥。 **注意** 对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 **ssh-agent** 进程使用的 SSH 密钥。	一个或多个密钥。例如： sshKey: <key1> <key2> <key3>

2.5.6.1.4. 可选的 AWS 配置参数

下表描述了可选的 AWS 配置参数：

表 2.8. 可选的 AWS 参数

参数	描述	值
compute.platform.aws.amiID	用于为集群引导计算机器的 AWS AMI。对于需要自定义 RHCOS AMI 的区域来说，这是必需的。	属于集合 AWS 区域的任何已发布或自定义 RHCOS AMI。
compute.platform.aws.rootVolume.iops	为根卷保留的每秒输入/输出操作 (IOPS) 数。	整数，如 **4000**。
compute.platform.aws.rootVolume.size	以 GiB 为单位的根卷大小。	整数，如 **500**。
compute.platform.aws.rootVolume.type	根卷的类型。	有效的 AWS EBS 卷类型，如 **io1**。
compute.platform.aws.rootVolume.kmsKeyARN	KMS 密钥的 Amazon 资源名称（密钥 ARN）。这是使用特定 KMS 密钥加密 worker 节点的操作系统卷。	有效的 密钥 ID 或密钥 ARN。
compute.platform.aws.type	计算机器的 EC2 实例类型。	有效的 AWS 实例类型，如 **c5.9xlarge**。
compute.platform.aws.zones	安装程序在其中为计算机机器池创建机器的可用区。如果您提供自己的 VPC，则必须在那个可用域中提供一个子网。	有效 AWS 可用区的列表，如 **us-east-1c**，以 YAML 序列表示。
compute.aws.region	安装程序在其中创建计算资源的 AWS 区域。	任何有效的 AWS 区域，如 **us-east-1**。
controlPlane.platform.aws.amiID	用于为集群引导 control plane 机器的 AWS AMI。对于需要自定义 RHCOS AMI 的区域来说，这是必需的。	属于集合 AWS 区域的任何已发布或自定义 RHCOS AMI。
controlPlane.platform.aws.rootVolume.kmsKeyARN	KMS 密钥的 Amazon 资源名称（密钥 ARN）。这需要使用特定的 KMS 密钥加密 control plane 节点的操作系统卷。	有效的 密钥 ID 和密钥 ARN。

参数	描述	值
controlPlane.platform.aws.type	control plane 机器的 EC2 实例类型。	有效的 AWS 实例类型，如 **c5.9xlarge**。
controlPlane.platform.aws.zones	安装程序在其中为 control plane 机器池创建机器的可用区。	有效 AWS 可用区的列表，如 **us-east-1c**，以 YAML 序列表示。
controlPlane.aws.region	安装程序在其中创建 control plane 资源的 AWS 区域。	有效的 AWS 区域，如 **us-east-1**。
platform.aws.amiID	用于为集群引导所有机器的 AWS AMI。如果设置，AMI 必须属于与集群相同的区域。对于需要自定义 RHCOS AMI 的区域来说，这是必需的。	属于集合 AWS 区域的任何已发布或自定义 RHCOS AMI。
platform.aws.serviceEndpoints.name	AWS 服务端点名称。只有在必须使用替代 AWS 端点（如 FIPS）时，才需要自定义端点。可以为 EC2、S3、IAM、Elastic Load Balancing、Tagging、Route 53 和 STS AWS 服务指定自定义 API 端点。	有效的 AWS 服务端点名称。
platform.aws.serviceEndpoints.url	AWS 服务端点 URL。URL 必须使用 **https** 协议，主机必须信任该证书。	有效的 AWS 服务端点 URL。
platform.aws.userTags	键与值的映射，安装程序将其作为标签添加到它所创建的所有资源。	任何有效的 YAML 映射，如 **<key>: <value>** 格式的键值对。如需有关 AWS 标签的更多信息，请参阅 AWS 文档中的标记您的 Amazon EC2 资源。
platform.aws.subnets	如果您提供 VPC，而不是让安装程序为您创建 VPC，请指定要使用的集群子网。子网必须是您指定的同一 **machineNetwork[].cidr** 范围的一部分。对于标准集群，为每个可用区指定一个公共和私有子网。对于私有集群，为每个可用区指定一个私有子网。	有效的子网 ID。

2.5.6.2. AWS 的自定义 install-config.yaml 文件示例

您可以自定义 **install-config.yaml** 文件，以指定有关 OpenShift Container Platform 集群平台的更多信息，或修改所需参数的值。

重要

此示例 YAML 文件仅供参考。您必须使用安装程序来获取 **install-config.yaml** 文件，并且修改该文件。

```
apiVersion: v1
baseDomain: example.com (1)
credentialsMode: Mint (2)
controlPlane: (3) (4)
  hyperthreading: Enabled (5)
  name: master
  platform:
    aws:
      zones:
      - us-west-2a
      - us-west-2b
      rootVolume:
        iops: 4000
        size: 500
        type: io1 (6)
      type: m5.xlarge
  replicas: 3
compute: (7)
- hyperthreading: Enabled (8)
  name: worker
  platform:
    aws:
      rootVolume:
        iops: 2000
        size: 500
        type: io1 (9)
      type: c5.4xlarge
      zones:
      - us-west-2c
  replicas: 3
metadata:
  name: test-cluster (10)
networking: (11)
  clusterNetwork:
  - cidr: 10.128.0.0/14
    hostPrefix: 23
  machineNetwork:
  - cidr: 10.0.0.0/16
  networkType: OpenShiftSDN
  serviceNetwork:
  - 172.30.0.0/16
platform:
  aws:
    region: us-west-2 (12)
    userTags:
      adminContact: jdoe
      costCenter: 7536
    amiID: ami-96c6f8f7 (13)
    serviceEndpoints: (14)
```

```
    - name: ec2
      url: https://vpce-id.ec2.us-west-2.vpce.amazonaws.com
fips: false (15)
sshKey: ssh-ed25519 AAAA... (16)
pullSecret: '{"auths": ...}' (17)
```

(1)(10)(12)(17)必需。安装程序会提示您输入这个值。

(2) 可选：添加此参数来强制 Cloud Credential Operator（CCO）使用指定的模式，而不是让 CCO 动态尝试决定凭证的功能。如需有关 CCO 模式的详情，请参阅 *Red Hat Operator* 参考内容中的 *Cloud Credential Operator* 条目。

(3)(7)(11)如果没有提供这些参数和值，安装程序会提供默认值。

(4) **controlPlane** 部分是一个单映射，但 **compute** 部分是一系列映射。为满足不同数据结构的要求，**compute** 部分的第一行必须以连字符 **-** 开头，**controlPlane** 部分的第一行则不可以连字符开头。只使用一个 control plane 池。

(5)(8)是否要启用或禁用并发多线程或**超线程**。默认情况下，启用并发多线程以提高机器内核的性能。您可以通过将参数值设为 **Disabled** 来禁用。如果您在某些集群机器上禁用并发多线程，则必须在所有集群机器上禁用。

重要

如果禁用并发多线程，请确保在容量规划时考虑到机器性能可能会显著降低的问题。如果您对机器禁用并发多线程，请使用较大的实例类型，如 **m4.2xlarge** 或 **m5.2xlarge**。

(6)(9)要为 etcd 配置更快的存储，特别是对于较大的集群，请将存储类型设置为 **io1**，并将 **iops** 设为 **2000**。

(13) 用于为集群引导机器的 AMI ID。如果设置，AMI 必须属于与集群相同的区域。

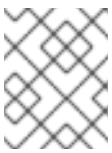
(14) AWS 服务端点。在安装到未知 AWS 区域时，需要自定义端点。端点 URL 必须使用 **https** 协议，主机必须信任该证书。

(15) 是否启用或禁用 FIPS 模式。默认情况下不启用 FIPS 模式。如果启用了 FIPS 模式，运行 OpenShift Container Platform 的 Red Hat Enterprise Linux CoreOS (RHCOS) 机器会绕过默认的 Kubernetes 加密套件，并使用由 RHCOS 提供的加密模块。

重要

只有在 **x86_64** 架构中的 OpenShift Container Platform 部署支持 FIPS 验证的/Modules in Process 加密库。

(16) 您可以选择提供您用来访问集群中机器的 **sshKey** 值。



注意

对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 **ssh-agent** 进程使用的 SSH 密钥。

2.5.6.3. 在安装过程中配置集群范围代理

生产环境可能会拒绝直接访问互联网，而是提供 HTTP 或 HTTPS 代理。您可以通过在 **install-config.yaml** 文件中配置代理设置，将新的 OpenShift Container Platform 集群配置为使用代理。

先决条件

- 您有一个现有的 **install-config.yaml** 文件。
- 您检查了集群需要访问的站点，并决定是否需要绕过代理。默认情况下代理所有集群出口流量，包括对托管云供应商 API 的调用。您需要将站点添加到 **Proxy** 对象的 **spec.noProxy** 字段来绕过代理。

注意

Proxy 对象 **status.noProxy** 字段使用安装配置中的 **networking.machineNetwork[].cidr**、**networking.clusterNetwork[].cidr** 和 **networking.serviceNetwork[]** 字段的值填充。

对于在 Amazon Web Services(AWS)、Google Cloud Platform(GCP)、Microsoft Azure 和 Red Hat OpenStack Platform(RHOSP)上安装, **Proxy** 对象 **status.noProxy** 字段也会使用实例元数据端点填充(**169.254.169.254**)。

- 如果您的集群位于 AWS 上，请将 **ec2.<region>.amazonaws.com**、**elasticloadbalancing.<region>.amazonaws.com** 和 **s3.<region>.amazonaws.com** 端点添加到 VPC 端点。需要这些端点才能完成节点到 AWS EC2 API 的请求。由于代理在容器级别而不是节点级别工作，因此您必须通过 AWS 专用网络将这些请求路由到 AWS EC2 API。在代理服务器中的允许列表中添加 EC2 API 的公共 IP 地址是不够的。

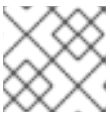
流程

1. 编辑 **install-config.yaml** 文件并添加代理设置。例如：

```
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: example.com 3
additionalTrustBundle: | 4
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
...
```

1. 用于创建集群外 HTTP 连接的代理 URL。URL 必须是 **http**。
2. 用于创建集群外 HTTPS 连接的代理 URL。
3. 要排除在代理中的目标域名、IP 地址或其他网络 CIDR 的逗号分隔列表。在域前面加 **.** 来仅匹配子域。例如： **.y.com** 匹配 **x.y.com**，但不匹配 **y.com**。使用 ***** 绕过所有目的地的代理。
4. 如果提供，安装程序会在 **openshift-config 命名空间中生成名为 user-ca- bundle** 的配置映射来保存额外的 CA 证书。如果您提供 **additionalTrustBundle** 和至少一个代理设置，则

Proxy 对象会被配置为引用 **trustedCA** 字段中的 **user-ca-bundle** 配置映射。然后，Cluster Network Operator 会创建一个 **trusted-ca-bundle** 配置映射，该配置映射将为 **trustedCA** 参数指定的内容与 RHCOS 信任捆绑包合并。**additionalTrustBundle** 字段是必需的，除非代理的身份证书由来自 RHCOS 信任捆绑包的颁发机构签名。



注意

安装程序不支持代理的 **readinessEndpoints** 字段。

2. 保存该文件，并在安装 OpenShift Container Platform 时引用。

安装程序会创建一个名为 **cluster** 的集群范围代理，该代理使用提供的 **install-config.yaml** 文件中的代理设置。如果没有提供代理设置，仍然会创建一个 **cluster Proxy** 对象，但它会有一个空 **spec**。

注意

只支持名为 **cluster** 的 **Proxy** 对象，且无法创建额外的代理。

2.5.7. Cluster Network Operator 配置

集群网络的配置作为 Cluster Network Operator (CNO) 配置的一部分被指定，并存储在名为 **cluster**的自定义资源（CR）对象中。CR 指定 **operator.openshift.io** API 组中的 **Network** API 的字段。

CNO 配置会在集群安装过程中从 **Network.config.openshift.io** API 组中的 **Network** API 继承以下字段，这些字段无法更改：

clusterNetwork

从中分配 pod IP 地址的 IP 地址池。

serviceNetwork

服务的 IP 地址池。

defaultNetwork.type

集群网络供应商，如 OpenShift SDN 或 OVN-Kubernetes。

您可以通过在名为 **cluster** 的 CNO 对象中设置 **defaultNetwork** 对象的字段来为集群指定集群网络供应商配置。

2.5.7.1. Cluster Network Operator 配置对象

Cluster Network Operator（CNO）的字段在下表中描述：

表 2.9. Cluster Network Operator 配置对象

字段	类型	描述
metadata.name	**字符串**	CNO 对象的名称。这个名称始终是 **cluster**。

字段	类型	描述
spec.clusterNetwork	**数组**	用于指定从哪些 IP 地址块分配 Pod IP 地址以及分配给集群中每个节点的子网前缀长度的列表。例如： spec: clusterNetwork: - cidr: 10.128.0.0/19 hostPrefix: 23 - cidr: 10.128.32.0/19 hostPrefix: 23 此值是只读的，并在 **install-config.yaml** 文件中指定。
spec.serviceNetwork	**数组**	服务的 IP 地址块。OpenShift SDN 和 OVN-Kubernetes Container Network Interface（CNI）网络供应商只支持服务网络具有单个 IP 地址块。例如： spec: serviceNetwork: - 172.30.0.0/14 此值是只读的，并在 **install-config.yaml** 文件中指定。
spec.defaultNetwork	**对象**	为集群网络配置 Container Network Interface（CNI）集群网络供应商。
spec.kubeProxyConfig	**对象**	此对象的字段指定 kube-proxy 配置。如果您使用 OVN-Kubernetes 集群网络供应商，则 kube-proxy 的配置不会起作用。

defaultNetwork 对象配置

defaultNetwork 对象的值在下表中定义：

表 2.10. **defaultNetwork** 对象

字段	类型	描述
type	**字符串**	**OpenShiftSDN** 或 **OVNKubernetes**。在安装过程中选择了集群网络供应商。集群安装后无法更改这个值。  **注意** OpenShift Container Platform 默认使用 OpenShift SDN Container Network Interface（CNI）集群网络供应商。

字段	类型	描述
openshiftSDNConfig	**对象**	此对象仅对 OpenShift SDN 集群网络供应商有效。
ovnKubernetesConfig	**对象**	此对象仅对 OVN-Kubernetes 集群网络供应商有效。

配置 OpenShift SDN CNI 集群网络供应商

下表描述了 OpenShift SDN Container Network Interface（CNI）集群网络供应商的配置字段。

表 2.11. openshiftSDNConfig 对象

字段	类型	描述
mode	**字符串**	配置 OpenShift SDN 的网络隔离模式。默认值为 **NetworkPolicy**。 **Multitenant** 和 **Subnet** 的值可以向后兼容 OpenShift Container Platform 3.x，但不推荐这样做。集群安装后无法更改这个值。
mtu	**整数**	VXLAN 覆盖网络的最大传输单元 (MTU) 。这根据主网络接口的 MTU 自动探测。您通常不需要覆盖检测到的 MTU。 如果自动探测的值不是您期望的，请确认节点上主网络接口中的 MTU 是正确的。您不能使用这个选项更改节点上主网络接口的 MTU 值。 如果您的集群中的不同节点需要不同的 MTU 值，则必须将此值设置为比集群中的最低 MTU 值小 **50**。例如，如果集群中的某些节点的 MTU 为 **9001**，而某些节点的 MTU 为 **1500**，则必须将此值设置为 **1450**。 集群安装后无法更改这个值。
vxlanPort	**整数**	用于所有 VXLAN 数据包的端口。默认值为 **4789**。集群安装后无法更改这个值。 如果您在虚拟环境中运行，并且现有节点是另一个 VXLAN 网络的一部分，那么可能需要更改此值。例如，当在 VMware NSX-T 上运行 OpenShift SDN 覆盖时，您必须为 VXLAN 选择一个备用端口，因为两个 SDN 都使用相同的默认 VXLAN 端口号。 在 Amazon Web Services (AWS) 上，您可以在端口 **9000** 和端口 **9999** 之间为 VXLAN 选择一个备用端口。

OpenShift SDN 配置示例

```
defaultNetwork:
  type: OpenShiftSDN
  openshiftSDNConfig:
```

```
    mode: NetworkPolicy
    mtu: 1450
    vxlanPort: 4789
```

配置 OVN-Kubernetes CNI 集群网络供应商
下表描述了 OVN-Kubernetes CNI 集群网络供应商的配置字段。

表 2.12. ovnKubernetesConfig 对象

字段	类型	描述
mtu	**整数**	Geneve（Generic Network Virtualization Encapsulation）覆盖网络的最大传输单元（MTU）。这根据主网络接口的 MTU 自动探测。您通常不需要覆盖检测到的 MTU。 如果自动探测的值不是您期望的，请确认节点上主网络接口中的 MTU 是正确的。您不能使用这个选项更改节点上主网络接口的 MTU 值。 如果您的集群中的不同节点需要不同的 MTU 值，则必须将此值设置为比集群中的最低 MTU 值小 **100**。例如，如果集群中的某些节点的 MTU 为 **9001**，而某些节点的 MTU 为 **1500**，则必须将此值设置为 **1400**。 集群安装后无法更改这个值。
genevePort	**整数**	用于所有 Geneve 数据包的端口。默认值为 **6081**。集群安装后无法更改这个值。

OVN-Kubernetes 配置示例

```
defaultNetwork:
  type: OVNKubernetes
  ovnKubernetesConfig:
    mtu: 1400
    genevePort: 6081
```

kubeProxyConfig 对象配置
kubeProxyConfig 对象的值在下表中定义：

表 2.13. kubeProxyConfig 对象

字段	类型	描述

字段	类型	描述
iptablesSyncPeriod	**字符串**	**iptables** 规则的刷新周期。默认值为 **30s**。有效的后缀包括 **s**、**m** 和 **h**，具体参见 Go **time** 软件包文档。 **注意** 由于 OpenShift Container Platform 4.3 及更高版本中引进了性能上的改进，现在不再需要调整 **iptablesSyncPeriod** 参数。
proxyArguments.iptables-min-sync-period	**数组**	刷新 **iptables** 规则前的最短时长。此字段确保刷新的频率不会过于频繁。有效的后缀包括 **s**、**m** 和 **h**，具体参见 Go **time** 软件包。默认值为： kubeProxyConfig: proxyArguments: iptables-min-sync-period: - 0s

2.5.8. 指定高级网络配置

您可以通过为集群网络供应商指定额外的配置，使用高级配置自定义将集群整合到现有网络环境中。您只能在安装集群前指定高级网络配置。

重要

不支持修改安装程序创建的 OpenShift Container Platform 清单文件。支持应用您创建的清单文件，如以下流程所示。

先决条件

- 创建 **install-config.yaml** 文件并完成对其所做的任何修改。

流程

1. 进入包含安装程序的目录并创建清单：

```
$ ./openshift-install create manifests --dir <installation_directory>
```

其中：

<installation_directory>
指定包含集群的 **install-config.yaml** 文件的目录名称。

2. 在 **<installation_directory>/manifests/** 目录下，为高级网络配置创建一个名为 **cluster-network-03-config.yml** 的 stub 清单文件：

```
$ cat <<EOF > <installation_directory>/manifests/cluster-network-03-config.yml
apiVersion: operator.openshift.io/v1
```

```
kind: Network
metadata:
  name: cluster
spec:
EOF
```

其中：

<installation_directory>

指定包含集群的 **manifests/** 目录的目录名称。

3. 在编辑器中打开 **cluster-network-03-config.yml** 文件，并为集群指定高级网络配置，如下例所示：

为 OpenShift SDN 网络供应商指定不同的 VXLAN 端口

```
apiVersion: operator.openshift.io/v1
kind: Network
metadata:
  name: cluster
spec:
  defaultNetwork:
    openshiftSDNConfig:
      vxlanPort: 4800
```

4. 保存 **cluster-network-03-config.yml** 文件，再退出文本编辑器。

5. 可选：备份 **manifests/cluster-network-03-config.yml** 文件。创建集群时，安装程序会删除 **manifests/** 目录。

注意

有关在 AWS 中使用网络负载平衡（Network Load Balancer）的更多信息，请参阅 使用网络负载平衡器在 AWS 上配置 Ingress 集群流量。

2.5.9. 在新 AWS 集群上配置 Ingress Controller 网络负载平衡

您可在新集群中创建一个由 AWS Network Load Balancer（NLB）支持的 Ingress Controller。

先决条件

- 创建 **install-config.yaml** 文件并完成对其所做的任何修改。

流程

在新集群中，创建一个由 AWS NLB 支持的 Ingress Controller。

1. 进入包含安装程序的目录并创建清单：

```
$ ./openshift-install create manifests --dir <installation_directory> 1
```

1 对于 **<installation_directory>**，请指定含有集群的 **install-config.yaml** 文件的目录的名称。

2. 在 **<installation_directory>/manifests/** 目录中创建一个名为 **cluster-ingress-default-ingresscontroller.yaml** 的文件：

```
$ touch <installation_directory>/manifests/cluster-ingress-default-ingresscontroller.yaml 1
```

1 对于 **<installation_directory>**，请指定包含集群的 **manifests/** 目录的目录名称。

创建该文件后，**manifests/** 目录中会包含多个网络配置文件，如下所示：

```
$ ls <installation_directory>/manifests/cluster-ingress-default-ingresscontroller.yaml
```

输出示例

```
cluster-ingress-default-ingresscontroller.yaml
```

3. 在编辑器中打开 **cluster-ingress-default-ingresscontroller.yaml** 文件，并输入描述您想要的 Operator 配置的 CR:

```
apiVersion: operator.openshift.io/v1
kind: IngressController
metadata:
  creationTimestamp: null
  name: default
  namespace: openshift-ingress-operator
spec:
  endpointPublishingStrategy:
    loadBalancer:
      scope: External
      providerParameters:
        type: AWS
        aws:
          type: NLB
    type: LoadBalancerService
```

4. 保存 **cluster-ingress-default-ingresscontroller.yaml** 文件并退出文本编辑器。

5. 可选：备份 **manifests/cluster-ingress-default-ingresscontroller.yaml** 文件。创建集群时，安装程序会删除 **manifests/** 目录。

2.5.10. 使用 OVN-Kubernetes 配置混合网络

您可以将集群配置为使用 OVN-Kubernetes 的混合网络。这允许支持不同节点网络配置的混合集群。例如：集群中运行 Linux 和 Windows 节点时需要这样做。

重要

您必须在安装集群过程中使用 OVN-Kubernetes 配置混合网络。您不能在安装过程中切换到混合网络。

先决条件

- 您在 **install-config.yaml** 文件中为 **networking.networkType** 参数定义了 **OVNKubernetes**。如需更多信息，请参阅有关在所选云供应商上配置 OpenShift Container Platform 网络自定义的安装文档。

流程

1. 进入包含安装程序的目录并创建清单：

```
$ ./openshift-install create manifests --dir <installation_directory>
```

其中：

<installation_directory>
指定包含集群的 **install-config.yaml** 文件的目录名称。

2. 在 **<installation_directory>/manifests/** 目录下，为高级网络配置创建一个名为 **cluster-network-03-config.yml** 的 stub 清单文件：

```
$ cat <<EOF > <installation_directory>/manifests/cluster-network-03-config.yml
apiVersion: operator.openshift.io/v1
kind: Network
metadata:
  name: cluster
spec:
EOF
```

其中：

<installation_directory>
指定包含集群的 **manifests/** 目录的目录名称。

3. 在编辑器中打开 **cluster-network-03-config.yml** 文件，并使用混合网络配置 OVN-Kubernetes，如下例所示：

指定混合网络配置

```
apiVersion: operator.openshift.io/v1
kind: Network
metadata:
  name: cluster
spec:
  defaultNetwork:
    ovnKubernetesConfig:
      hybridOverlayConfig:
        hybridClusterNetwork: 1
        - cidr: 10.132.0.0/14
          hostPrefix: 23
        hybridOverlayVXLANPort: 9898 2
```

1 指定用于额外覆盖网络上节点的 CIDR 配置。**hybridClusterNetwork** CIDR 无法与 **clusterNetwork** CIDR 重叠。

2 为额外覆盖网络指定自定义 VXLAN 端口。这是在 vSphere 上安装的集群中运行 Windows

4. 保存 **cluster-network-03-config.yml** 文件，再退出文本编辑器。
5. 可选：备份 **manifests/cluster-network-03-config.yml** 文件。创建集群时，安装程序会删除 **manifests/** 目录。

注意

有关在同一集群中使用 Linux 和 Windows 节点的更多信息，请参阅了解 Windows 容器工作负载。

2.5.11. 部署集群

您可以在兼容云平台中安装 OpenShift Container Platform。

重要

安装程序的 **create cluster** 命令只能在初始安装过程中运行一次。

先决条件

- 配置托管集群的云平台的帐户。
- 获取 OpenShift Container Platform 安装程序以及集群的 pull secret。

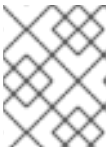
流程

1. 更改为包含安装程序的目录并初始化集群部署：

```
$ ./openshift-install create cluster --dir <installation_directory> \ 1
    --log-level=info 2
```

1 对于 **<installation_directory>**，请指定自定义 **./install-config.yaml** 文件的位置。

2 要查看不同的安装详情，请指定 **warn**、**debug** 或 **error**，而不要指定 **info**。



注意

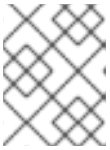
如果您在主机上配置的云供应商帐户没有足够的权限来部署集群，安装过程将会停止，并且显示缺少的权限。

集群部署完成后，终端会显示访问集群的信息，包括指向其 Web 控制台的链接和 **kubeadmin** 用户的凭证。

输出示例

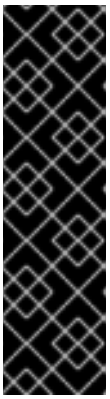
```
...
INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run 'export
KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'
INFO Access the OpenShift web-console here: https://console-openshift-
console.apps.mycluster.example.com
```

```
INFO Login to the console with user: "kubeadmin", and password: "4vYBz-Ee6gm-ymBZj-Wt5AL"
INFO Time elapsed: 36m22s
```



注意

当安装成功时，集群访问和凭证信息还会输出到 **<installation_directory>/.openshift_install.log**。



重要

- 安装程序生成的 Ignition 配置文件包含在 24 小时后过期的证书，然后在过期时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外情况是，您需要手动批准待处理的 **node-bootstrapper** 证书签名请求（CSR）来恢复 kubelet 证书。如需更多信息，请参阅*从过期的 control plane 证书中恢复*的文档。
- 建议您在生成 12 小时后使用 Ignition 配置文件，因为集群安装后 24 小时证书从 16 小时轮转至 22 小时。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中运行证书更新时避免安装失败。

重要

您不得删除安装程序或安装程序所创建的文件。需要这两者才能删除集群。

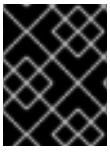
2. 可选：从您用来安装集群的 IAM 帐户删除或禁用 **AdministratorAccess** 策略。

注意

只有在安装过程中才需要 **AdministratorAccess** 策略提供的升级权限。

2.5.12. 通过下载二进制文件安装 OpenShift CLI

您需要安装 CLI（**oc**） 来使用命令行界面与 OpenShift Container Platform 进行交互。您可在 Linux、Windows 或 macOS 上安装 **oc**。



重要

如果安装了旧版本的 **oc**，则无法使用 OpenShift Container Platform 4.6 中的所有命令。下载并安装新版本的 **oc**。

2.5.12.1. 在 Linux 上安装 OpenShift CLI

您可以按照以下流程在 Linux 上安装 OpenShift CLI（**oc**）二进制文件。

流程

1. 进入到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 在 **Version** 下拉菜单中选择相应的版本。
3. 单击 **OpenShift v4.6 Linux 客户端** 条目旁边的 **Download Now**，再保存文件。

4. 解包存档：

```
$ tar xvzf <file>
```

5. 把 **oc** 二进制代码放到 **PATH** 中的目录中。
执行以下命令可以查看当前的 **PATH** 设置：

```
$ echo $PATH
```

安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
$ oc <command>
```

2.5.12.2. 在 Windows 上安装 OpenShift CLI

您可以按照以下流程在 Windows 上安装 OpenShift CLI（**oc**）二进制代码。

流程

1. 进入到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 在 **Version** 下拉菜单中选择相应的版本。
3. 单击 **OpenShift v4.6 Windows 客户端**条目旁边的 **Download Now**，再保存文件。
4. 使用 ZIP 程序解压存档。
5. 把 **oc** 二进制代码放到 **PATH** 中的目录中。
要查看您的 **PATH**，请打开命令提示窗口并执行以下命令：

```
C:\> path
```

安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
C:\> oc <command>
```

2.5.12.3. 在 macOS 上安装 OpenShift CLI

您可以按照以下流程在 macOS 上安装 OpenShift CLI（**oc**）二进制代码。

流程

1. 进入到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 在 **Version** 下拉菜单中选择相应的版本。
3. 单击 **OpenShift v4.6 MacOSX 客户端**条目旁边的 **Download Now**，再保存文件。
4. 解包和解压存档。
5. 将 **oc** 二进制文件移到 PATH 的目录中。
要查看您的 **PATH**，打开一个终端窗口并执行以下命令：

```
$ echo $PATH
```

安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
$ oc <command>
```

2.5.13. 使用 CLI 登录到集群

您可以通过导出集群 **kubeconfig** 文件，以默认系统用户身份登录集群。**kubeconfig** 文件包含关于集群的信息，供 CLI 用于将客户端连接到正确集群和 API 服务器。该文件特只适用于一个特定的集群，在 OpenShift Container Platform 安装过程中创建。

先决条件

- 已部署了 OpenShift Container Platform 集群。
- 已安装 **oc** CLI。

流程

1. 导出 **kubeadmin** 凭证：

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

2. 使用导出的配置，验证能否成功运行 **oc** 命令：

```
$ oc whoami
```

输出示例

```
system:admin
```

2.5.14. 使用 Web 控制台登录到集群

kubeadmin 用户默认在 OpenShift Container Platform 安装后存在。您可以使用 OpenShift Container Platform Web 控制台以 **kubeadmin** 用户身份登录集群。

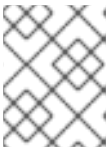
先决条件

- 有访问安装主机的访问权限。
- 您完成了集群安装，所有集群 Operator 都可用。

流程

1. 从安装主机上的 **kubeadmin -password** 文件中获取 kubeadmin 用户的密码：

```
$ cat <installation_directory>/auth/kubeadmin-password
```



注意

另外，您还可以从安装主机上的 **<installation_directory>/.openshift_install.log** 日志文件获取 **kubeadmin** 密码。

2. 列出 OpenShift Container Platform Web 控制台路由：

```
$ oc get routes -n openshift-console | grep 'console-openshift'
```

注意

另外，您还可以从安装主机上的 **<installation_directory>/.openshift_install.log 日志** 文件获取 OpenShift Container Platform 路由。

输出示例

```
console     console-openshift-console.apps.<cluster_name>.<base_domain>           console
https   reencrypt/Redirect   None
```

3. 在 Web 浏览器中导航到上一命令输出中包括的路由，以 **kubeadmin** 用户身份登录。

其他资源

- 如需有关访问和了解 OpenShift Container Platform Web 控制台的更多信息，请参阅访问 Web 控制台。

2.5.15. OpenShift Container Platform 的 Telemetry 访问

在 OpenShift Container Platform 4.6 中，默认运行的 Telemetry 服务提供有关集群健康状况和成功更新的指标，需要访问互联网。如果您的集群连接到互联网，Telemetry 会自动运行，而且集群会注册到 OpenShift Cluster Manager。

确认 OpenShift Cluster Manager 清单正确后，可以由 Telemetry 自动维护，也可以使用 OpenShift Cluster Manager 手动维护，使用订阅监控来跟踪帐户或多集群级别的 OpenShift Container Platform 订阅。

其他资源

- 有关 Telemetry 服务的更多信息，请参阅关于远程健康监控。

2.5.16. 后续步骤

- 验证安装。
- 自定义集群。
- 如果需要，您可以选择不使用远程健康报告。
- 如果需要，您可以删除云供应商凭证。

2.6. 在受限网络中的 AWS 上安装集群

在 OpenShift Container Platform 版本 4.6 中，您可以通过在现有 Amazon Virtual Private Cloud（VPC）上创建安装发行内容的内部镜像在受限网络中的 Amazon Web Services（AWS）上安装集群。

2.6.1. 先决条件

- 您已为断开连接的安装 mirror 了 registry 的镜像 并获取您的 OpenShift Container Platform 版本的 **imageContentSources** 数据。

重要

由于安装介质位于堡垒主机上，因此请使用该计算机完成所有安装步骤。

- AWS 中有一个现有的 VPC。当使用安装程序置备的基础架构安装到受限网络时，无法使用安装程序置备的 VPC。您必须使用用户置备的 VPC 来满足以下要求之一：
  - 包含镜像 registry。
  - 具有防火墙规则或对等连接来访问其他位置托管的镜像 registry。
- 您可以参阅有关 OpenShift Container Platform 安装和更新流程 的详细信息。
- 已将 AWS 帐户配置为托管集群。

重要

如果您的计算机上存储有 AWS 配置集，则不要在使用多因素验证设备的同时使用您生成的临时会话令牌。在集群的整个生命周期中，集群会持续使用您的当前 AWS 凭证来创建 AWS 资源，因此您必须使用基于密钥的长期凭证。要生成适当的密钥，请参阅 AWS 文档中的管理 IAM 用户的访问密钥。您可在运行安装程序时提供密钥。

- 您下载了 AWS CLI 并安装到您的计算机上。请参阅 AWS 文档中的使用捆绑安装程序（Linux、macOS 或 Unix）安装 AWS CLI。
- 如果使用防火墙并计划使用 Telemetry 服务，需要 将防火墙配置为允许集群需要访问的站点。

注意

如果要配置代理，请务必还要查看此站点列表。

- 如果不允许系统管理身份和访问管理（IAM），集群管理员可以 手动创建和维护 IAM 凭证。手动模式也可以用于云 IAM API 无法访问的环境中。

2.6.2. 关于在受限网络中安装

在 OpenShift Container Platform 4.6 中，可以执行不需要有效的互联网连接来获取软件组件的安装。受限网络安装可使用安装程序置备的基础架构或用户置备的基础架构完成，具体取决于您要安装集群的云平台。

如果选择在云平台中执行受限网络安装，仍然需要访问其云 API。有些云功能，比如 Amazon Web Service 的 Route 53 DNS 和 IAM 服务，需要访问互联网。根据您的网络，在裸机硬件或 VMware vSphere 上安装时可能需要较少的互联网访问。

要完成受限网络安装，您必须创建一个 registry，镜像 OpenShift Container Platform registry 的内容并包含其安装介质。您可以在堡垒主机上创建此镜像，该主机可同时访问互联网和您的封闭网络，也可以使用满足您的限制条件的其他方法。

2.6.2.1. 其他限制

受限网络中的集群还有以下额外限制：

- **ClusterVersion** 状态包含一个 **Unable to retrieve available updates** 错误。
- 默认情况下，您无法使用 Developer Catalog 的内容，因为您无法访问所需的镜像流标签。

2.6.3. 关于使用自定义 VPC

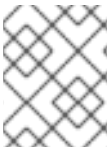
在 OpenShift Container Platform 4.6 中，您可以在 Amazon Web Services（AWS）的现有 Amazon Virtual Private Cloud（VPC）中将集群部署到现有子网中。通过将 OpenShift Container Platform 部署到现有的 AWS VPC 中，您可能会避开新帐户中的限制，或者更容易地利用公司所设置的操作限制。如果您无法获得您自己创建 VPC 所需的基础架构创建权限，请使用这个安装选项。

因为安装程序无法了解您现有子网中还有哪些其他组件，所以无法选择子网 CIDR 。您必须为安装集群的子网配置网络。

2.6.3.1. 使用 VPC 的要求

安装程序不再创建以下组件：

- 互联网网关
- NAT 网关
- 子网
- 路由表
- VPCs
- VPC DHCP 选项
- VPC 端点



注意

安装程序要求您使用由云提供的 DNS 服务器。不支持使用自定义 DNS 服务器，并导致安装失败。

如果您使用自定义 VPC，您必须为安装程序和集群正确配置它及其子网。有关创建和管理 AWS VPC VPC 的更多信息，请参阅 AWS 文档中的 Amazon VPC 控制台向导配置 以及处理 VPC 和子网。

安装程序无法：

- 子类网络范围供群集使用。
- 设置子网的路由表。
- 设置 VPC 选项，如 DHCP。

您必须在安装集群前完成这些任务。有关在 AWS VPC 中配置网络的更多信息，请参阅 VPC 的 VPC 网络组件 和路由表。

您的 VPC 必须满足以下特征：

- VPC 不能使用 **kubernetes.io/cluster/.*: owned** 标签。
  安装程序会修改子网以添加 **kubernetes.io/cluster/.*: shared** 标签，因此您的子网必须至少有一个可用的空闲标签插槽。请参阅 AWS 文档中的 标签限制 部分，以确认安装程序可以为您指定的每个子网添加标签。

- 您必须在 VPC **中启用 enable DnsSupport 和 enableDnsHostnames** 属性，以便集群可以使用附加到 VPC 的 Route 53 区域来解析集群内部 DNS 记录。请参阅 AWS 文档中的您的 VPC 中的 DNS 支持部分 。
  如果您希望使用自己的 Route 53 托管私有区，则必须在安装集群前将现有托管区与 VPC 关联。您可以使用 **install-config.yaml** 文件中的 **platform.aws.hostedZone** 字段定义托管区。

- 如果您使用具有公共访问权限的集群，您必须为每个集群使用的可用区创建一个公共和私有子网。每个可用区不能包含多于一个的公共子网和私有子网。

如果您在断开连接的环境中工作，您将无法访问 EC2 和 ELB 端点的公共 IP 地址。要解决这个问题，您必须创建一个 VPC 端点，并将其附加到集群使用的子网。端点应命名如下：

- **ec2.<region>.amazonaws.com**
- **elasticloadbalancing.<region>.amazonaws.com**
- **s3.<region>.amazonaws.com**

所需的 VPC 组件

您必须提供合适的 VPC 和子网，以便与您的机器通信。

组件	AWS 类型	描述
VPC	• **AWS::EC2::VPC** • **AWS::EC2::VPCEndpoint**	您必须提供一个公共 VPC 供集群使用。VPC 使用引用每个子网的路由表的端点，以改进与托管在 S3 中的 registry 的通信。
公共子网	• **AWS::EC2::Subnet** • **AWS::EC2::SubnetNetworkAclAssociation**	您的 VPC 必须有 1 到 3 个可用区的公共子网，并将其与适当的入口规则关联。

<table>
<tr><th>组件</th><th>AWS 类型</th><th colspan="2">描述</th></tr>
<tr><td>互联网网关</td><td>AWS::EC2::InternetGateway
AWS::EC2::VPCGatewayAttachment
AWS::EC2::RouteTable
AWS::EC2::Route
AWS::EC2::SubnetRouteTableAssociation
AWS::EC2::NatGateway
AWS::EC2::EIP</td><td colspan="2">您必须有一个公共互联网网关，以及附加到 VPC 的公共路由。在提供的模板中，每个公共子网都有一个具有 EIP 地址的 NAT 网关。这些 NAT 网关允许集群资源（如专用子网实例）访问互联网，而有些受限网络或代理场景则不需要它们。</td></tr>
<tr><td>网络访问控制</td><td>AWS::EC2::NetworkAcl
AWS::EC2::NetworkAclEntry</td><td colspan="2">您必须允许 VPC 访问下列端口：</td></tr>
<tr><td></td><td></td><td>端口</td><td>原因</td></tr>
<tr><td></td><td></td><td>80</td><td>入站 HTTP 流量</td></tr>
<tr><td></td><td></td><td>443</td><td>入站 HTTPS 流量</td></tr>
<tr><td></td><td></td><td>22</td><td>入站 SSH 流量</td></tr>
<tr><td></td><td></td><td>1024 - 65535</td><td>入站临时流量</td></tr>
<tr><td></td><td></td><td>0 - 65535</td><td>出站临时流量</td></tr>
<tr><td>专用子网</td><td>AWS::EC2::Subnet
AWS::EC2::RouteTable
AWS::EC2::SubnetRouteTableAssociation</td><td colspan="2">您的 VPC 可以具有私有子网。提供的 CloudFormation 模板可为 1 到 3 个可用区创建专用子网。如果您使用专用子网，必须为其提供适当的路由和表。</td></tr>
</table>

2.6.3.2. VPC 验证

要确保您提供的子网适合您的环境，安装程序会确认以下信息：

- 您指定的所有子网都存在。
- 您提供了私有子网。
- 子网 CIDR 属于您指定的机器 CIDR。
- 您为每个可用区提供子网。每个可用区不包含多于一个的公共子网和私有子网。如果您使用私有集群，为每个可用区只提供一个私有子网。否则，为每个可用区提供一个公共和私有子网。

- 您可以为每个私有子网可用区提供一个公共子网。机器不会在没有为其提供私有子网的可用区中置备。

如果您销毁使用现有 VPC 的集群，VPC 不会被删除。从 VPC 中删除 OpenShift Container Platform 集群时，**kubernetes.io/cluster/.*: shared** 标签会从使用它的子网中删除。

2.6.3.3. 权限划分

从 OpenShift Container Platform 4.3 开始，您不需要安装程序置备的基础架构集群部署所需的所有权限。这与您所在机构可能已有的权限划分类似：不同的个人可以在您的云中创建不同的资源。。例如，您可以创建针对于特定应用程序的对象，如实例、存储桶和负载均衡器，但不能创建与网络相关的组件，如 VPC 、子网或入站规则。

您在创建集群时使用的 AWS 凭证不需要 VPC 和 VPC 中的核心网络组件（如子网、路由表、互联网网关、NAT 和 VPN）所需的网络权限。您仍然需要获取集群中的机器需要的应用程序资源的权限，如 ELB 、安全组、S3 存储桶和节点。

2.6.3.4. 集群间隔离

如果您将 OpenShift Container Platform 部署到现有网络中，集群服务的隔离将在以下方面减少：

- 您可以在同一 VPC 中安装多个 OpenShift Container Platform 集群。
- 整个网络允许 ICMP 入站流量。
- 整个网络都允许 TCP 22 入站流量 (SSH)。
- 整个网络都允许 control plane TCP 6443 入站流量 (Kubernetes API)。
- 整个网络都允许 control plane TCP 22623 入站流量 (MCS) 。

2.6.4. OpenShift Container Platform 的互联网访问

在 OpenShift Container Platform 4.6 中，您需要访问互联网来获得用来安装集群的镜像。

您必须具有以下互联网访问权限：

- 访问 OpenShift Cluster Manager 以下载安装程序并执行订阅管理。如果集群可以访问互联网，并且没有禁用 Telemetry，该服务会自动授权您的集群。
- 访问 Quay.io，以获取安装集群所需的软件包。
- 获取执行集群更新所需的软件包。

重要

如果您的集群无法直接访问互联网，则可以在置备的某些类基础架构上执行受限网络安装。在此过程中，您要下载所需的内容，并使用它在镜像 registry（mirror registry） 中填充安装集群并生成安装程序所需的软件包。对于某些安装类型，集群要安装到的环境不需要访问互联网。在更新集群之前，要更新 registry 镜像系统中的内容。

2.6.5. 生成 SSH 私钥并将其添加到代理中

如果要在集群上执行安装调试或灾难恢复，则必须为 **ssh-agent** 和安装程序提供 SSH 密钥。您可以使用此密钥访问公共集群中的 bootstrap 机器来排除安装问题。

注意

在生产环境中，您需要进行灾难恢复和调试。

您可以使用此密钥以 **core** 用户身份通过 SSH 连接到 master 节点。在部署集群时，此密钥会添加到 **core** 用户的 **~/.ssh/authorized_keys** 列表中。

注意

您必须使用一个本地密钥，而不要使用在特定平台上配置的密钥，如 AWS 密钥对。

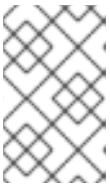
流程

1. 如果还没有为计算机上免密码身份验证而配置的 SSH 密钥，请创建一个。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ ssh-keygen -t ed25519 -N '' \
  -f <path>/<file_name> 1
```

1 指定新 SSH 密钥的路径和文件名，如 **~/.ssh/id_rsa**。如果您已有密钥对，请确保您的公钥位于 **~/.ssh** 目录中。

运行此命令会在指定的位置生成不需要密码的 SSH 密钥。



注意

如果您计划在 **x86_64** 架构中安装使用 FIPS 验证的/Modules in Process 加密库的 OpenShift Container Platform 集群，不要创建使用 **ed25519** 算法的密钥。反之，创建一个使用 **rsa** 或 **ecdsa** 算法的密钥。

2. 作为后台任务启动 **ssh-agent** 进程：

```
$ eval "$(ssh-agent -s)"
```

输出示例

```
Agent pid 31874
```

注意

如果您的集群采用 FIPS 模式，则只使用 FIPS 兼容算法来生成 SSH 密钥。密钥必须是 RSA 或 ECDSA。

3. 将 SSH 私钥添加到 **ssh-agent**：

```
$ ssh-add <path>/<file_name> 1
```

输出示例

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

1 指定 SSH 私钥的路径和文件名，如 **~/.ssh/id_rsa**

后续步骤

- 在安装 OpenShift Container Platform 时，为安装程序提供 SSH 公钥。

2.6.6. 创建安装配置文件

您可以自定义在 Amazon Web Services (AWS) 上安装的 OpenShift Container Platform 集群。

先决条件

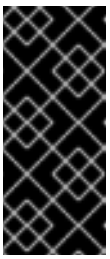
- 获取 OpenShift Container Platform 安装程序以及集群的 pull secret。对于受限网络安装，这些文件位于您的堡垒主机上。
- 具有创建镜像容器镜像仓库（registry）时生成的 **imageContentSources** 值。
- 获取您的镜像 registry 的证书内容。

流程

1. 创建 **install-config.yaml** 文件。
   a. 更改到包含安装程序的目录，再运行以下命令：

```
$ ./openshift-install create install-config --dir <installation_directory> 1
```

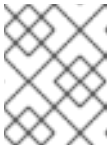
1 对于 **<installation_directory>**，请指定用于保存安装程序所创建的文件的目录名称。



重要

指定一个空目录。一些安装信息，如 bootstrap X.509 证书，有较短的过期间隔，因此不要重复使用安装目录。如果要重复使用另一个集群安装中的个别文件，可以将其复制到您的目录中。但是，一些安装数据的文件名可能会在发行版本之间有所改变。从 OpenShift Container Platform 老版本中复制安装文件时要格外小心。

   b. 在提示符处，提供您的云的配置详情：
      i. 可选：选择用来访问集群机器的 SSH 密钥。



注意

对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 **ssh-agent** 进程使用的 SSH 密钥。

      ii. 选择 **AWS** 作为目标平台。
      iii. 如果计算机上没有保存 Amazon Web Services (AWS) 配置集，请为您配置用于运行安装程序的用户输入 AWS 访问密钥 ID 和 Secret 访问密钥。
      iv. 选择要将集群部署到的 AWS 区域。

v. 选择您为集群配置的 Route 53 服务的基域。

vi. 为集群输入一个描述性名称。

vii. 粘贴 Red Hat OpenShift Cluster Manager 中的 pull secret 。

2. 编辑 **install-config.yaml** 文件，以提供在受限网络中安装所需的其他信息。

a. 更新 **pullSecret** 值，使其包含 registry 的身份验证信息：

```
pullSecret: '{"auths":{"<mirror_host_name>:5000": {"auth": "<credentials>","email":
"you@example.com"}}}'
```

对于 **<mirror_host_name>**，请指定您在镜像 registry 证书中指定的 registry 域名；对于 **<credentials>**，请指定您的镜像 registry 的 base64 编码用户名和密码。

b. 添加 **additionalTrustBundle** 参数和值。

```
additionalTrustBundle: |
  -----BEGIN CERTIFICATE-----

ZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZ
  -----END CERTIFICATE-----
```

该值必须是您用于镜像 registry 的证书文件内容，可以是现有的可信证书颁发机构或您为镜像 registry 生成的自签名证书。

c. 在以下位置定义 VPC 安装集群的子网：

```
subnets:
- subnet-1
- subnet-2
- subnet-3
```

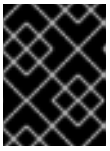
d. 添加镜像内容资源，如下例所示：

```
imageContentSources:
- mirrors:
  - <mirror_host_name>:5000/<repo_name>/release
  source: quay.example.com/openshift-release-dev/ocp-release
- mirrors:
  - <mirror_host_name>:5000/<repo_name>/release
  source: registry.example.com/ocp/release
```

要完成这些值，请使用您在创建镜像容器镜像仓库（registry）时记录的 **imageContentSources**。

3. 对需要的 **install-config.yaml** 文件做任何其他修改。您可以在**安装配置参数**部分中找到有关可用参数的更多信息。

4. 备份 **install-config.yaml** 文件，以便用于安装多个集群。



重要

install-config.yaml 文件会在安装过程中消耗掉。如果要重复使用此文件，必须现在备份。

2.6.6.1. 安装配置参数

在部署 OpenShift Container Platform 集群前，您可以提供参数值，以描述托管集群的云平台的帐户并选择性地自定义集群平台。在创建 **install-config.yaml** 安装配置文件时，您可以通过命令行来提供所需的参数的值。如果要自定义集群，可以修改 **install-config.yaml** 文件来提供关于平台的更多信息。

注意

安装之后，您无法修改 **install-config.yaml** 文件中的这些参数。

重要

openshift-install 命令不验证参数的字段名称。如果指定了不正确的名称，则不会创建相关的文件或对象，且不会报告错误。确保所有指定的参数的字段名称都正确。

2.6.6.1.1. 所需的配置参数

下表描述了所需的安装配置参数：

表 2.14. 所需的参数

参数	描述	值
apiVersion	**install-config.yaml** 内容的 API 版本。当前版本是 **v1**。安装程序还可能支持旧的 API 版本。	字符串
baseDomain	云供应商的基域。此基础域用于创建到 OpenShift Container Platform 集群组件的路由。集群的完整 DNS 名称是 **baseDomain** 和 **metadata.name** 参数值的组合，其格式为 **<metadata.name>.<baseDomain>**。	完全限定域名或子域名，如 **example.com**。
metadata	Kubernetes 资源 **ObjectMeta**，其中只消耗 **name** 参数。	对象
metadata.name	集群的名称。集群的 DNS 记录是 **{{.metadata.name}}.{{.baseDomain}}** 的子域。	小写字母,连字符(-)和句点(.)的字符串，如 **dev**。

参数	描述	值
platform	执行安装的具体平台配置：**aws**、**baremetal**、**azure**、**openstack**、**ovirt**、**vsphere**。有关 **platform.<platform>** 参数的额外信息，请参考下表来了解您的具体平台。	对象

pullSecret	从 Red Hat OpenShift Cluster Manager 获取 pull secret，验证从 Quay.io 等服务中下载 OpenShift Container Platform 组件的容器镜像。	<pre>{
		  "auths":{
		  "cloud.openshift.com":{
		  "auth":"b3Blb=",
		  "email":"you@example.com"
		  },
		  "quay.io":{
		  "auth":"b3Blb=",
		  "email":"you@example.com"
		  }
		  }
		}</pre>

2.6.6.1.2. 网络配置参数

您可以根据现有网络基础架构的要求自定义安装配置。例如，您可以扩展集群网络的 IP 地址块，或者提供不同于默认值的不同 IP 地址块。

只支持 IPv4 地址。

表 2.15. 网络参数

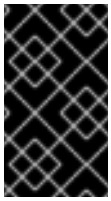
参数	描述	值
networking	集群网络的配置。	对象 **注意** 您不能在安装后修改 **networking** 对象指定的参数。
networking.networkType	要安装的集群网络供应商 Container Network Interface（CNI）插件。	**OpenShiftSDN** 或 **OVNKubernetes**。默认值为 **OpenShiftSDN**。

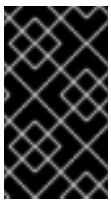
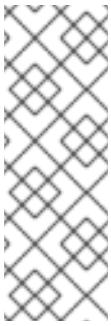
参数	描述	值
networking.clusterNetwork	pod 的 IP 地址块。 默认值为 **10.128.0.0/14**，主机前缀为 **/23**。 如果您指定多个 IP 地址块，则块不得互相重叠。	一个对象数组。例如： networking: clusterNetwork: - cidr: 10.128.0.0/14 hostPrefix: 23
networking.clusterNetwork.cidr	使用 **networking.clusterNetwork** 时需要此项。IP 地址块。 一个 IPv4 网络。	使用 CIDR 形式的 IP 地址块。IPv4 块的前缀长度介于 **0** 到 **32** 之间。
networking.clusterNetwork.hostPrefix	分配给每个单独节点的子网前缀长度。例如，如果 **hostPrefix** 设为 **23**，则每个节点从所给的 **cidr** 中分配一个 **/23** 子网。**hostPrefix** 值 **23** 提供 510（2^(32 - 23)- 2）个 pod IP 地址。	子网前缀。 默认值为 **23**。
networking.serviceNetwork	服务的 IP 地址块。默认值为 **172.30.0.0/16**。 OpenShift SDN 和 OVN-Kubernetes 网络供应商只支持服务网络的一个 IP 地址块。	CIDR 格式具有 IP 地址块的数组。例如： networking: serviceNetwork: - 172.30.0.0/16
networking.machineNetwork	机器的 IP 地址块。 如果您指定多个 IP 地址块，则块不得互相重叠。	一个对象数组。例如： networking: machineNetwork: - cidr: 10.0.0.0/16
networking.machineNetwork.cidr	使用 **networking.machineNetwork** 时需要。IP 地址块。libvirt 以外的所有平台的默认值为 **10.0.0.0/16**。对于 libvirt，默认值为 **192.168.126.0/24**。	CIDR 表示法中的 IP 网络块。 例如：**10.0.0.0/16**。 **注意** 将 **networking.machineNetwork** 设置为与首选 NIC 所在的 CIDR 匹配。

2.6.6.1.3. 可选配置参数

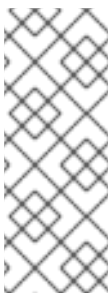
下表描述了可选安装配置参数：

表 2.16. 可选参数

参数	描述	值
additionalTrustBundle	添加到节点可信证书存储中的 PEM 编码 X.509 证书捆绑包。配置了代理时，也可以使用这个信任捆绑包。	字符串
compute	组成计算节点的机器的配置。	machine-pool 对象的数组。详情请查看以下"Machine-pool"表。
compute.architecture	决定池中机器的指令集合架构。目前不支持异构集群，因此所有池都必须指定相同的架构。有效值为 **amd64** （默认值）。	字符串
compute.hyperthreading	是否在计算机器上启用或禁用并发多线程或**超线程**。默认情况下，启用并发多线程以提高机器内核的性能。  **重要** 如果禁用并发多线程，请确保在容量规划时考虑到机器性能可能会显著降低的问题。	**Enabled** 或 **Disabled**
compute.name	使用 **compute** 时需要此值。机器池的名称。	**worker**
compute.platform	使用 **compute** 时需要此值。使用此参数指定托管 worker 机器的云供应商。此参数值必须与 **controlPlane.platform** 参数值匹配。	**aws**、**azure**、**gcp**、**openstack**、**ovirt**、**vsphere** 或 **{}**
compute.replicas	要置备的计算机器数量，也称为 worker 机器。	大于或等于 **2** 的正整数。默认值为 **3**。
controlPlane	组成 control plane 的机器的配置。	**MachinePool** 对象的数组。详情请查看以下"Machine-pool"表。
controlPlane.architecture	决定池中机器的指令集合架构。目前不支持异构集群，因此所有池都必须指定相同的架构。有效值为 **amd64** （默认值）。	字符串

参数	描述	值
controlPlane.hyperthreading	是否在 control plane 机器上启用或禁用并发多线程或**超线程**。默认情况下，启用并发多线程以提高机器内核的性能。  **重要** 如果禁用并发多线程，请确保在容量规划时考虑到机器性能可能会显著降低的问题。	**Enabled** 或 **Disabled**
controlPlane.name	使用 **controlPlane** 时需要。机器池的名称。	**master**
controlPlane.platform	使用 **controlPlane** 时需要。使用此参数指定托管 control plane 机器的云供应商。此参数值必须与 **compute.platform** 参数值匹配。	**aws**、**azure**、**gcp**、**openstack**、**ovirt**、**vsphere** 或 **{}**
controlPlane.replicas	要置备的 control plane 机器数量。	唯一支持的值是 **3**，它是默认值。
credentialsMode	Cloud Credential Operator（CCO）模式。如果没有指定任何模式，CCO 会动态地尝试决定提供的凭证的功能，在支持多个模式的平台上使用 mint 模式。  **注意** 不是所有 CCO 模式都支持所有云供应商。如需有关 CCO 模式的更多信息，请参阅*Red Hat Operator 参考指南*内容中的 *Cloud Credential Operator* 条目。	**Mint**、**Passthrough**、**Manual** 或空字符串(**""**)。

参数	描述	值
fips	启用或禁用 FIPS 模式。默认为 **false**（禁用）。如果启用了 FIPS 模式，运行 OpenShift Container Platform 的 Red Hat Enterprise Linux CoreOS (RHCOS) 机器会绕过默认的 Kubernetes 加密套件，并使用由 RHCOS 提供的加密模块。  **重要** 只有在 **x86_64** 架构中的 OpenShift Container Platform 部署支持 FIPS 验证的/Modules in Process 加密库。  **注意** 如果使用 Azure File 存储，则无法启用 FIPS 模式。	**false** 或 **true**
imageContentSources	release-image 内容的源和仓库。	对象数组。包括一个 **source** 以及可选的 **mirrors**，如下表所示。
imageContentSources.source	使用 **imageContentSources** 时需要。指定用户在镜像拉取规格中引用的仓库。	字符串
imageContentSources.mirrors	指定可能还包含同一镜像的一个或多个仓库。	字符串数组
publish	如何发布或公开集群的面向用户的端点，如 Kubernetes API、OpenShift 路由。	**Internal** 或 **External**。把 **publish** 设置为 **Internal** 以部署一个私有集群，它不能被互联网访问。默认值为 **External**。

sshKey	用于验证集群机器访问的 SSH 密钥或密钥。
	
	 
	注意
	
	对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 **ssh-agent** 进程使用的 SSH 密钥。
		一个或多个密钥。例如：
		
		`sshKey:`
		`  <key1>`
		`  <key2>`
		`  <key3>`

2.6.6.1.4. 可选的 AWS 配置参数

下表描述了可选的 AWS 配置参数：

表 2.17. 可选的 AWS 参数

参数	描述	值
compute.platform.aws.amiID	用于为集群引导计算机器的 AWS AMI。对于需要自定义 RHCOS AMI 的区域来说，这是必需的。	属于集合 AWS 区域的任何已发布或自定义 RHCOS AMI。
compute.platform.aws.rootVolume.iops	为根卷保留的每秒输入/输出操作 (IOPS) 数。	整数，如 **4000**。
compute.platform.aws.rootVolume.size	以 GiB 为单位的根卷大小。	整数，如 **500**。
compute.platform.aws.rootVolume.type	根卷的类型。	有效的 AWS EBS 卷类型，如 **io1**。
compute.platform.aws.rootVolume.kmsKeyARN	KMS 密钥的 Amazon 资源名称（密钥 ARN）。这是使用特定 KMS 密钥加密 worker 节点的操作系统卷。	有效的 密钥 ID 或密钥 ARN。
compute.platform.aws.type	计算机器的 EC2 实例类型。	有效的 AWS 实例类型，如 **c5.9xlarge**。
compute.platform.aws.zones	安装程序在其中为计算机机器池创建机器的可用区。如果您提供自己的 VPC，则必须在那个可用域中提供一个子网。	有效 AWS 可用区的列表，如 **us-east-1c**，以 YAML 序列表示。
compute.aws.region	安装程序在其中创建计算资源的 AWS 区域。	任何有效的 AWS 区域，如 **us-east-1**。
controlPlane.platform.aws.amiID	用于为集群引导 control plane 机器的 AWS AMI。对于需要自定义 RHCOS AMI 的区域来说，这是必需的。	属于集合 AWS 区域的任何已发布或自定义 RHCOS AMI。
controlPlane.platform.aws.rootVolume.kmsKeyARN	KMS 密钥的 Amazon 资源名称（密钥 ARN）。这需要使用特定的 KMS 密钥加密 control plane 节点的操作系统卷。	有效的 密钥 ID 和密钥 ARN。

参数	描述	值
controlPlane.platform.aws.type	control plane 机器的 EC2 实例类型。	有效的 AWS 实例类型，如 **c5.9xlarge**。
controlPlane.platform.aws.zones	安装程序在其中为 control plane 机器池创建机器的可用区。	有效 AWS 可用区的列表，如 **us-east-1c**，以 YAML 序列表示。
controlPlane.aws.region	安装程序在其中创建 control plane 资源的 AWS 区域。	有效的 AWS 区域，如 **us-east-1**。
platform.aws.amiID	用于为集群引导所有机器的 AWS AMI。如果设置，AMI 必须属于与集群相同的区域。对于需要自定义 RHCOS AMI 的区域来说，这是必需的。	属于集合 AWS 区域的任何已发布或自定义 RHCOS AMI。
platform.aws.serviceEndpoints.name	AWS 服务端点名称。只有在必须使用替代 AWS 端点（如 FIPS）时，才需要自定义端点。可以为 EC2、S3、IAM、Elastic Load Balancing、Tagging、Route 53 和 STS AWS 服务指定自定义 API 端点。	有效的 AWS 服务端点名称。
platform.aws.serviceEndpoints.url	AWS 服务端点 URL。URL 必须使用 **https** 协议，主机必须信任该证书。	有效的 AWS 服务端点 URL。
platform.aws.userTags	键与值的映射，安装程序将其作为标签添加到它所创建的所有资源。	任何有效的 YAML 映射，如 **<key>: <value>** 格式的键值对。如需有关 AWS 标签的更多信息，请参阅 AWS 文档中的标记您的 Amazon EC2 资源。
platform.aws.subnets	如果您提供 VPC，而不是让安装程序为您创建 VPC，请指定要使用的集群子网。子网必须是您指定的同一 **machineNetwork[].cidr** 范围的一部分。对于标准集群，为每个可用区指定一个公共和私有子网。对于私有集群，为每个可用区指定一个私有子网。	有效的子网 ID。

2.6.6.2. AWS 的自定义 install-config.yaml 文件示例

您可以自定义 **install-config.yaml** 文件，以指定有关 OpenShift Container Platform 集群平台的更多信息，或修改所需参数的值。

重要

此示例 YAML 文件仅供参考。您必须使用安装程序来获取 **install-config.yaml** 文件，并且修改该文件。

```
apiVersion: v1
baseDomain: example.com (1)
credentialsMode: Mint (2)
controlPlane: (3) (4)
  hyperthreading: Enabled (5)
  name: master
  platform:
    aws:
      zones:
      - us-west-2a
      - us-west-2b
      rootVolume:
        iops: 4000
        size: 500
        type: io1 (6)
      type: m5.xlarge
  replicas: 3
compute: (7)
- hyperthreading: Enabled (8)
  name: worker
  platform:
    aws:
      rootVolume:
        iops: 2000
        size: 500
        type: io1 (9)
      type: c5.4xlarge
      zones:
      - us-west-2c
  replicas: 3
metadata:
  name: test-cluster (10)
networking:
  clusterNetwork:
  - cidr: 10.128.0.0/14
    hostPrefix: 23
  machineNetwork:
  - cidr: 10.0.0.0/16
  networkType: OpenShiftSDN
  serviceNetwork:
  - 172.30.0.0/16
platform:
  aws:
    region: us-west-2 (11)
    userTags:
      adminContact: jdoe
      costCenter: 7536
    subnets: (12)
    - subnet-1
```

```
    - subnet-2
    - subnet-3
    amiID: ami-96c6f8f7 13
    serviceEndpoints: 14
      - name: ec2
        url: https://vpce-id.ec2.us-west-2.vpce.amazonaws.com
    hostedZone: Z3URY6TWQ91KVV 15
fips: false 16
sshKey: ssh-ed25519 AAAA... 17
pullSecret: '{"auths":{"<local_registry>": {"auth": "<credentials>","email": "you@example.com"}}}' 18
additionalTrustBundle: | 19
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
imageContentSources: 20
- mirrors:
  - <local_registry>/<local_repository_name>/release
  source: quay.io/openshift-release-dev/ocp-release
- mirrors:
  - <local_registry>/<local_repository_name>/release
  source: quay.io/openshift-release-dev/ocp-v4.0-art-dev
```

1 10 11 必需。安装程序会提示您输入这个值。

2 可选：添加此参数来强制 Cloud Credential Operator（CCO）使用指定的模式，而不是让 CCO 动态尝试决定凭证的功能。如需有关 CCO 模式的详情，请参阅 *Red Hat Operator* 参考内容中的 *Cloud Credential Operator* 条目。

3 7 如果没有提供这些参数和值，安装程序会提供默认值。

4 **controlPlane** 部分是一个单映射，但 **compute** 部分是一系列映射。为满足不同数据结构的要求，**compute** 部分的第一行必须以连字符 **-** 开头，**controlPlane** 部分的第一行则不可以连字符开头。只使用一个 control plane 池。

5 8 是否要启用或禁用并发多线程或**超线程**。默认情况下，启用并发多线程以提高机器内核的性能。您可以通过将参数值设为 **Disabled** 来禁用。如果您在某些集群机器上禁用并发多线程，则必须在所有集群机器上禁用。

重要

如果禁用并发多线程，请确保在容量规划时考虑到机器性能可能会显著降低的问题。如果您对机器禁用并发多线程，请使用较大的实例类型，如 **m4.2xlarge** 或 **m5.2xlarge**。

6 9 要为 etcd 配置更快的存储，特别是对于较大的集群，请将存储类型设置为 **io1**，并将 **iops** 设为 **2000**。

12 如果您提供自己的 VPC，为集群使用的每个可用区指定子网。

13 用于为集群引导机器的 AMI ID。如果设置，AMI 必须属于与集群相同的区域。

14 AWS 服务端点。在安装到未知 AWS 区域时，需要自定义端点。端点 URL 必须使用 **https** 协议，主机必须信任该证书。

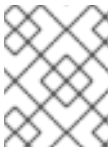
15 您现有 Route 53 私有托管区的 ID。提供现有的托管区需要您提供自己的 VPC，托管区已在安装集群前与 VPC 关联。如果未定义，安装程序会创建一个新的托管区。

16 是否启用或禁用 FIPS 模式。默认情况下不启用 FIPS 模式。如果启用了 FIPS 模式，运行 OpenShift Container Platform 的 Red Hat Enterprise Linux CoreOS (RHCOS) 机器会绕过默认的 Kubernetes 加密套件，并使用由 RHCOS 提供的加密模块。

重要

只有在 **x86_64** 架构中的 OpenShift Container Platform 部署支持 FIPS 验证的/Modules in Process 加密库。

17 您可以选择提供您用来访问集群中机器的 **sshKey** 值。



注意

对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 **ssh-agent** 进程使用的 SSH 密钥。

18 对于 **<local_registry>**，请指定 registry 域名，以及您的镜像 registry 用来提供内容的可选端口。例如： **registry.example.com** 或者 **registry.example.com:5000**。使用 **<credentials>** 为您生成的镜像 registry 指定 base64 编码的用户名和密码。

19 提供用于镜像 registry 的证书文件内容。

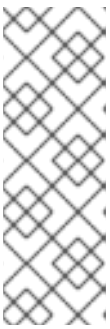
20 提供命令输出中的 **imageContentSources** 部分来镜像存储库。

2.6.6.3. 在安装过程中配置集群范围代理

生产环境可能会拒绝直接访问互联网，而是提供 HTTP 或 HTTPS 代理。您可以通过在 **install-config.yaml** 文件中配置代理设置，将新的 OpenShift Container Platform 集群配置为使用代理。

先决条件

- 您有一个现有的 **install-config.yaml** 文件。
- 您检查了集群需要访问的站点，并决定是否需要绕过代理。默认情况下代理所有集群出口流量，包括对托管云供应商 API 的调用。您需要将站点添加到 **Proxy** 对象的 **spec.noProxy** 字段来绕过代理。



注意

Proxy 对象 **status.noProxy** 字段使用安装配置中的 **networking.machineNetwork[].cidr**、**networking.clusterNetwork[].cidr** 和 **networking.serviceNetwork[]** 字段的值填充。

对于在 Amazon Web Services(AWS)、Google Cloud Platform(GCP)、Microsoft Azure 和 Red Hat OpenStack Platform(RHOSP)上安装, **Proxy** 对象 **status.noProxy** 字段也会使用实例元数据端点填充(**169.254.169.254**)。

- 如果您的集群位于 AWS 上，请将 **ec2.<region>.amazonaws.com**、**elasticloadbalancing.<region>.amazonaws.com** 和 **s3.<region>.amazonaws.com** 端点添加到 VPC 端点。需要这些端点才能完成节点到 AWS EC2 API 的请求。由于代理在容器级别而不是节点级别工作，因此您必

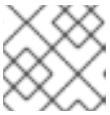
须通过 AWS 专用网络将这些请求路由到 AWS EC2 API。在代理服务器中的允许列表中添加 EC2 API 的公共 IP 地址是不够的。

流程

1. 编辑 **install-config.yaml** 文件并添加代理设置。例如：

```
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: example.com 3
additionalTrustBundle: | 4
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
...
```

1. 用于创建集群外 HTTP 连接的代理 URL。URL 必须是 **http**。
2. 用于创建集群外 HTTPS 连接的代理 URL。
3. 要排除在代理中的目标域名、IP 地址或其他网络 CIDR 的逗号分隔列表。在域前面加 **.** 来仅匹配子域。例如： **.y.com** 匹配 **x.y.com**，但不匹配 **y.com**。使用 ***** 绕过所有目的地的代理。
4. 如果提供，安装程序会在 **openshift-config 命名空间中生成名为 user-ca- bundle** 的配置映射来保存额外的 CA 证书。如果您提供 **additionalTrustBundle** 和至少一个代理设置，则 **Proxy** 对象会被配置为引用 **trustedCA** 字段中的 **user-ca-bundle** 配置映射。然后，Cluster Network Operator 会创建一个 **trusted-ca-bundle** 配置映射，该配置映射将为 **trustedCA** 参数指定的内容与 RHCOS 信任捆绑包合并。 **additionalTrustBundle** 字段是必需的，除非代理的身份证书由来自 RHCOS 信任捆绑包的颁发机构签名。

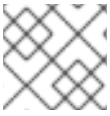


注意

安装程序不支持代理的 **readinessEndpoints** 字段。

2. 保存该文件，并在安装 OpenShift Container Platform 时引用。

安装程序会创建一个名为 **cluster** 的集群范围代理，该代理使用提供的 **install-config.yaml** 文件中的代理设置。如果没有提供代理设置，仍然会创建一个 **cluster Proxy** 对象，但它会有一个空 **spec**。



注意

只支持名为 **cluster** 的 **Proxy** 对象，且无法创建额外的代理。

2.6.7. 部署集群

您可以在兼容云平台中安装 OpenShift Container Platform。

重要

安装程序的 **create cluster** 命令只能在初始安装过程中运行一次。

先决条件

- 配置托管集群的云平台的帐户。
- 获取 OpenShift Container Platform 安装程序以及集群的 pull secret。

流程

1. 更改为包含安装程序的目录并初始化集群部署：

```
$ ./openshift-install create cluster --dir <installation_directory> \ 1
    --log-level=info 2
```

1 对于 **<installation_directory>**，请指定自定义 **./install-config.yaml** 文件的位置。

2 要查看不同的安装详情，请指定 **warn**、**debug** 或 **error**，而不要指定 **info**。

注意

如果您在主机上配置的云供应商帐户没有足够的权限来部署集群，安装过程将会停止，并且显示缺少的权限。

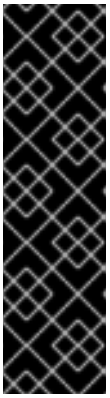
集群部署完成后，终端会显示访问集群的信息，包括指向其 Web 控制台的链接和 **kubeadmin** 用户的凭证。

输出示例

```
...
INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run 'export
KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'
INFO Access the OpenShift web-console here: https://console-openshift-
console.apps.mycluster.example.com
INFO Login to the console with user: "kubeadmin", and password: "4vYBz-Ee6gm-ymBZj-
Wt5AL"
INFO Time elapsed: 36m22s
```

注意

当安装成功时，集群访问和凭证信息还会输出到 **<installation_directory>/.openshift_install.log**。



重要

- 安装程序生成的 Ignition 配置文件包含在 24 小时后过期的证书，然后在过期时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外情况是，您需要手动批准待处理的 **node-bootstrapper** 证书签名请求（CSR）来恢复 kubelet 证书。如需更多信息，请参阅*从过期的 control plane 证书中恢复*的文档。
- 建议您在生成 12 小时后使用 Ignition 配置文件，因为集群安装后 24 小时证书从 16 小时轮转至 22 小时。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中运行证书更新时避免安装失败。

重要

您不得删除安装程序或安装程序所创建的文件。需要这两者才能删除集群。

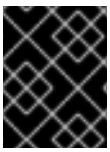
2. 可选：从您用来安装集群的 IAM 帐户删除或禁用 **AdministratorAccess** 策略。

注意

只有在安装过程中才需要 **AdministratorAccess** 策略提供的升级权限。

2.6.8. 通过下载二进制文件安装 OpenShift CLI

您需要安装 CLI（**oc**） 来使用命令行界面与 OpenShift Container Platform 进行交互。您可在 Linux、Windows 或 macOS 上安装 **oc**。



重要

如果安装了旧版本的 **oc**，则无法使用 OpenShift Container Platform 4.6 中的所有命令。下载并安装新版本的 **oc**。

2.6.8.1. 在 Linux 上安装 OpenShift CLI

您可以按照以下流程在 Linux 上安装 OpenShift CLI（**oc**）二进制文件。

流程

1. 进入到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 在 **Version** 下拉菜单中选择相应的版本。
3. 单击 **OpenShift v4.6 Linux 客户端** 条目旁边的 **Download Now**，再保存文件。
4. 解包存档：

```
$ tar xvzf <file>
```

5. 把 **oc** 二进制代码放到 **PATH** 中的目录中。
执行以下命令可以查看当前的 **PATH** 设置：

```
$ echo $PATH
```

安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
$ oc <command>
```

2.6.8.2. 在 Windows 上安装 OpenShift CLI

您可以按照以下流程在 Windows 上安装 OpenShift CLI（**oc**）二进制代码。

流程

1. 进入到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 在 **Version** 下拉菜单中选择相应的版本。
3. 单击 **OpenShift v4.6 Windows 客户端**条目旁边的 **Download Now**，再保存文件。
4. 使用 ZIP 程序解压存档。
5. 把 **oc** 二进制代码放到 **PATH** 中的目录中。
   要查看您的 **PATH**，请打开命令提示窗口并执行以下命令：

   ```
   C:\> path
   ```

安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
C:\> oc <command>
```

2.6.8.3. 在 macOS 上安装 OpenShift CLI

您可以按照以下流程在 macOS 上安装 OpenShift CLI（**oc**）二进制代码。

流程

1. 进入到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 在 **Version** 下拉菜单中选择相应的版本。
3. 单击 **OpenShift v4.6 MacOSX 客户端**条目旁边的 **Download Now**，再保存文件。
4. 解包和解压存档。
5. 将 **oc** 二进制文件移到 PATH 的目录中。
   要查看您的 **PATH**，打开一个终端窗口并执行以下命令：

   ```
   $ echo $PATH
   ```

安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
$ oc <command>
```

2.6.9. 使用 CLI 登录到集群

您可以通过导出集群 **kubeconfig** 文件，以默认系统用户身份登录集群。**kubeconfig** 文件包含关于集群的信息，供 CLI 用于将客户端连接到正确集群和 API 服务器。该文件特只适用于一个特定的集群，在 OpenShift Container Platform 安装过程中创建。

先决条件

- 已部署了 OpenShift Container Platform 集群。
- 已安装 **oc** CLI。

流程

1. 导出 **kubeadmin** 凭证：

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

2. 使用导出的配置，验证能否成功运行 **oc** 命令：

```
$ oc whoami
```

输出示例

```
system:admin
```

2.6.10. 禁用默认的 OperatorHub 源

在 OpenShift Container Platform 安装过程中，默认为 OperatorHub 配置由红帽和社区项目提供的源内容的 operator 目录。在受限网络环境中，必须以集群管理员身份禁用默认目录。

流程

- 通过在 **OperatorHub** 对象中添加 **disableAllDefaultSources: true** 来禁用默认目录的源：

```
$ oc patch OperatorHub cluster --type json \
    -p '[{"op": "add", "path": "/spec/disableAllDefaultSources", "value": true}]'
```

提示

或者，您可以使用 Web 控制台管理目录源。在 **Administration → Cluster Settings → Global Configuration → OperatorHub** 页面中，点 **Sources** 选项卡，其中可创建、删除、禁用和启用单独的源。

2.6.11. OpenShift Container Platform 的 Telemetry 访问

在 OpenShift Container Platform 4.6 中，默认运行的 Telemetry 服务提供有关集群健康状况和成功更新的指标，需要访问互联网。如果您的集群连接到互联网，Telemetry 会自动运行，而且集群会注册到 OpenShift Cluster Manager。

确认 OpenShift Cluster Manager 清单正确后，可以由 Telemetry 自动维护，也可以使用 OpenShift Cluster Manager 手动维护，使用订阅监控来跟踪帐户或多集群级别的 OpenShift Container Platform 订阅。

其他资源

- 有关 Telemetry 服务的更多信息，请参阅关于远程健康监控。

2.6.12. 后续步骤

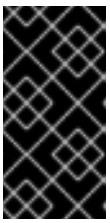
- 验证安装。
- 自定义集群。
- 为 Cluster Samples Operator 和 **must-gather** 工具配置镜像流。
- 了解如何在受限网络中使用 Operator Lifecycle Manager（OLM）。
- 如果您用来安装集群的镜像 registry 具有一个可信任的 CA，通过配置额外的信任存储将其添加到集群中。
- 如果需要，您可以选择不使用远程健康报告。

2.7. 在 AWS 上将集群安装到现有的 VPC 中

在 OpenShift Container Platform 版本 4.6 中，您可以在 Amazon Web Services（AWS）上将集群安装到现有 Amazon Virtual Private Cloud（VPC）中。安装程序会置备所需基础架构的其余部分，您可以进一步定制这些基础架构。要自定义安装，请在安装集群前修改 **install-config.yaml** 文件中的参数。

2.7.1. 先决条件

- 查看有关 OpenShift Container Platform 安装和更新流程的详细信息。
- 配置 AWS 帐户以托管集群。



重要

如果您的计算机上存储有 AWS 配置集，则不要在使用多因素验证设备的同时使用您生成的临时会话令牌。在集群的整个生命周期中，集群会持续使用您的当前 AWS 凭证来创建 AWS 资源，因此您必须使用长期凭证。要生成适当的密钥，请参阅 AWS 文档中的管理 IAM 用户的访问密钥。您可在运行安装程序时提供密钥。

- 如果使用防火墙，则必须将其配置为允许集群需要访问的站点。
- 如果不允许系统管理身份和访问管理（IAM），集群管理员可以 手动创建和维护 IAM 凭证。手动模式也可以用于云 IAM API 无法访问的环境中。

2.7.2. 关于使用自定义 VPC

在 OpenShift Container Platform 4.6 中，您可以在 Amazon Web Services（AWS）的现有 Amazon Virtual Private Cloud（VPC）中将集群部署到现有子网中。通过将 OpenShift Container Platform 部署到现有的 AWS VPC 中，您可能会避开新帐户中的限制，或者更容易地利用公司所设置的操作限制。如果您无法获得您自己创建 VPC 所需的基础架构创建权限，请使用这个安装选项。

因为安装程序无法了解您现有子网中还有哪些其他组件，所以无法选择子网 CIDR 。您必须为安装集群的子网配置网络。

2.7.2.1. 使用 VPC 的要求

安装程序不再创建以下组件：

- 互联网网关
- NAT 网关
- 子网
- 路由表
- VPCs
- VPC DHCP 选项
- VPC 端点

注意

安装程序要求您使用由云提供的 DNS 服务器。不支持使用自定义 DNS 服务器，并导致安装失败。

如果您使用自定义 VPC，您必须为安装程序和集群正确配置它及其子网。有关创建和管理 AWS VPC VPC 的更多信息，请参阅 AWS 文档中的 Amazon VPC 控制台向导配置 以及处理 VPC 和子网。

安装程序无法：

- 子类网络范围供群集使用。
- 设置子网的路由表。
- 设置 VPC 选项，如 DHCP。

您必须在安装集群前完成这些任务。有关在 AWS VPC 中配置网络的更多信息，请参阅 VPC 的 VPC 网络组件 和路由表。

您的 VPC 必须满足以下特征：

- 为集群使用的每个可用区创建一个公共和私有子网。每个可用区不能包含多于一个的公共子网和私有子网。有关此类配置的示例，请参阅 AWS 文档中的 具有公共和私有子网(NAT)的 VPC。记录每个子网 ID。完成安装要求您在 **install-config.yaml** 文件的 **platform** 部分输入这些值。请参阅 AWS 文档中的 查找子网 ID。
- VPC 的 CIDR 块必须包含 **Networking.machineCIDR** ，它是集群机器的 IP 地址池。子网 CIDR 块必须属于您指定的机器 CIDR。
- VPC 必须附加一个公共互联网网关。对于每个可用区：
  - 公共子网需要路由到互联网网关。
  - 公共子网需要一个具有 EIP 地址的 NAT 网关。
  - 专用子网需要路由到公共子网中的 NAT 网关。

- VPC 不能使用 **kubernetes.io/cluster/.*: owned** 标签。
  安装程序会修改子网以添加 **kubernetes.io/cluster/.*: shared** 标签，因此您的子网必须至少有一个可用的空闲标签插槽。请参阅 AWS 文档中的 标签限制 部分，以确认安装程序可以为您指定的每个子网添加标签。

- 您必须在 VPC 中启用 **enable DnsSupport 和 enableDnsHostnames** 属性，以便集群可以使用附加到 VPC 的 Route 53 区域来解析集群内部 DNS 记录。请参阅 AWS 文档中的您的 VPC 中的 DNS 支持部分 。
  如果您希望使用自己的 Route 53 托管私有区，则必须在安装集群前将现有托管区与 VPC 关联。您可以使用 **install-config.yaml** 文件中的 **platform.aws.hostedZone** 字段定义托管区。

如果您在断开连接的环境中工作，您将无法访问 EC2 和 ELB 端点的公共 IP 地址。要解决这个问题，您必须创建一个 VPC 端点，并将其附加到集群使用的子网。端点应命名如下：

- **ec2.<region>.amazonaws.com**
- **elasticloadbalancing.<region>.amazonaws.com**
- **s3.<region>.amazonaws.com**

所需的 VPC 组件

您必须提供合适的 VPC 和子网，以便与您的机器通信。

组件	AWS 类型	描述
VPC	• **AWS::EC2::VPC** • **AWS::EC2::VPCEndpoint**	您必须提供一个公共 VPC 供集群使用。VPC 使用引用每个子网的路由表的端点，以改进与托管在 S3 中的 registry 的通信。
公共子网	• **AWS::EC2::Subnet** • **AWS::EC2::SubnetNetworkAclAssociation**	您的 VPC 必须有 1 到 3 个可用区的公共子网，并将其与适当的入口规则关联。
互联网网关	• **AWS::EC2::InternetGateway** • **AWS::EC2::VPCGatewayAttachment** • **AWS::EC2::RouteTable** • **AWS::EC2::Route** • **AWS::EC2::SubnetRouteTableAssociation** • **AWS::EC2::NatGateway** • **AWS::EC2::EIP**	您必须有一个公共互联网网关，以及附加到 VPC 的公共路由。在提供的模板中，每个公共子网都有一个具有 EIP 地址的 NAT 网关。这些 NAT 网关允许集群资源（如专用子网实例）访问互联网，而有些受限网络或代理场景则不需要它们。

<table>
<tr><th>组件</th><th>AWS 类型</th><th colspan="2">描述</th></tr>
<tr><td rowspan="7">网络访问控制</td><td rowspan="7">AWS::EC2::NetworkAcl
AWS::EC2::NetworkAclEntry</td><td colspan="2">您必须允许 VPC 访问下列端口：</td></tr>
<tr><td>端口</td><td>原因</td></tr>
<tr><td>80</td><td>入站 HTTP 流量</td></tr>
<tr><td>443</td><td>入站 HTTPS 流量</td></tr>
<tr><td>22</td><td>入站 SSH 流量</td></tr>
<tr><td>1024 - 65535</td><td>入站临时流量</td></tr>
<tr><td>0 - 65535</td><td>出站临时流量</td></tr>
<tr><td>专用子网</td><td>AWS::EC2::Subnet
AWS::EC2::RouteTable
AWS::EC2::SubnetRouteTableAssociation</td><td colspan="2">您的 VPC 可以具有私有子网。提供的 CloudFormation 模板可为 1 到 3 个可用区创建专用子网。如果您使用专用子网，必须为其提供适当的路由和表。</td></tr>
</table>

2.7.2.2. VPC 验证

要确保您提供的子网适合您的环境，安装程序会确认以下信息：

- 您指定的所有子网都存在。
- 您提供了私有子网。
- 子网 CIDR 属于您指定的机器 CIDR。
- 您为每个可用区提供子网。每个可用区不包含多于一个的公共子网和私有子网。如果您使用私有集群，为每个可用区只提供一个私有子网。否则，为每个可用区提供一个公共和私有子网。
- 您可以为每个私有子网可用区提供一个公共子网。机器不会在没有为其提供私有子网的可用区中置备。

如果您销毁使用现有 VPC 的集群，VPC 不会被删除。从 VPC 中删除 OpenShift Container Platform 集群时，**kubernetes.io/cluster/.*: shared** 标签会从使用它的子网中删除。

2.7.2.3. 权限划分

从 OpenShift Container Platform 4.3 开始，您不需要安装程序置备的基础架构集群部署所需的所有权限。这与您所在机构可能已有的权限划分类似：不同的个人可以在您的云中创建不同的资源。。例如，您可以创建针对于特定应用程序的对象，如实例、存储桶和负载均衡器，但不能创建与网络相关的组件，如 VPC 、子网或入站规则。

您在创建集群时使用的 AWS 凭证不需要 VPC 和 VPC 中的其他网络组件（如子网、路由表、互联网网

您在创建集群时使用的 AWS 凭证不需要 VPC 和 VPC 中的核心网络组件（如子网、路由表、互联网网关、NAT 和 VPN）所需的网络权限。您仍然需要获取集群中的机器需要的应用程序资源的权限，如 ELB、安全组、S3 存储桶和节点。

2.7.2.4. 集群间隔离

如果您将 OpenShift Container Platform 部署到现有网络中，集群服务的隔离将在以下方面减少：

- 您可以在同一 VPC 中安装多个 OpenShift Container Platform 集群。
- 整个网络允许 ICMP 入站流量。
- 整个网络都允许 TCP 22 入站流量 (SSH)。
- 整个网络都允许 control plane TCP 6443 入站流量 (Kubernetes API)。
- 整个网络都允许 control plane TCP 22623 入站流量 (MCS) 。

2.7.3. OpenShift Container Platform 的互联网访问

在 OpenShift Container Platform 4.6 中，您需要访问互联网来安装集群。

您必须具有以下互联网访问权限：

- 访问 OpenShift Cluster Manager 以下载安装程序并执行订阅管理。如果集群可以访问互联网，并且没有禁用 Telemetry，该服务会自动授权您的集群。
- 访问 Quay.io，以获取安装集群所需的软件包。
- 获取执行集群更新所需的软件包。

重要

如果您的集群无法直接访问互联网，则可以在置备的某些类基础架构上执行受限网络安装。在此过程中，您要下载所需的内容，并使用它在镜像 registry（mirror registry） 中填充安装集群并生成安装程序所需的软件包。对于某些安装类型，集群要安装到的环境不需要访问互联网。在更新集群之前，要更新 registry 镜像系统中的内容。

2.7.4. 生成 SSH 私钥并将其添加到代理中

如果要在集群上执行安装调试或灾难恢复，则必须为 **ssh-agent** 和安装程序提供 SSH 密钥。您可以使用此密钥访问公共集群中的 bootstrap 机器来排除安装问题。

注意

在生产环境中，您需要进行灾难恢复和调试。

您可以使用此密钥以 **core** 用户身份通过 SSH 连接到 master 节点。在部署集群时，此密钥会添加到 **core** 用户的 **~/.ssh/authorized_keys** 列表中。

注意

您必须使用一个本地密钥，而不要使用在特定平台上配置的密钥，如 AWS 密钥对。

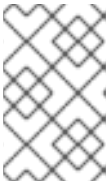
流程

1. 如果还没有为计算机上免密码身份验证而配置的 SSH 密钥，请创建一个。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ ssh-keygen -t ed25519 -N '' \
  -f <path>/<file_name> 1
```

1 指定新 SSH 密钥的路径和文件名，如 **~/.ssh/id_rsa**。如果您已有密钥对，请确保您的公钥位于 **~/.ssh** 目录中。

运行此命令会在指定的位置生成不需要密码的 SSH 密钥。



注意

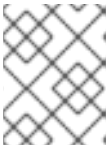
如果您计划在 **x86_64** 架构中安装使用 FIPS 验证的/Modules in Process 加密库的 OpenShift Container Platform 集群，不要创建使用 **ed25519** 算法的密钥。反之，创建一个使用 **rsa** 或 **ecdsa** 算法的密钥。

2. 作为后台任务启动 **ssh-agent** 进程：

```
$ eval "$(ssh-agent -s)"
```

输出示例

```
Agent pid 31874
```



注意

如果您的集群采用 FIPS 模式，则只使用 FIPS 兼容算法来生成 SSH 密钥。密钥必须是 RSA 或 ECDSA。

3. 将 SSH 私钥添加到 **ssh-agent**：

```
$ ssh-add <path>/<file_name> 1
```

输出示例

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

1 指定 SSH 私钥的路径和文件名，如 **~/.ssh/id_rsa**

后续步骤

- 在安装 OpenShift Container Platform 时，为安装程序提供 SSH 公钥。

2.7.5. 获取安装程序

在安装 OpenShift Container Platform 之前，将安装文件下载到本地计算机上。

先决条件

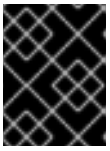
- 运行 Linux 或 macOS 的计算机，本地磁盘空间为 500 MB

流程

1. 访问 OpenShift Cluster Manager 站点的 Infrastructure Provider 页面。如果您有红帽帐号，请使用自己的凭证登录。如果没有，请创建一个帐户。
2. 选择您的基础架构供应商。
3. 进入适用于您的安装类型的页面，下载您的操作系统的安装程序，并将文件放在要保存安装配置文件的目录中。。

重要

安装程序会在用来安装集群的计算机上创建若干文件。在完成集群安装后，您必须保留安装程序和安装程序所创建的文件。这两个文件都需要删除集群。



重要

删除安装程序创建的文件不会删除您的集群，即使集群在安装过程中失败也是如此。要删除集群，为特定云供应商完成 OpenShift Container Platform 卸载流程。

4. 提取安装程序。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ tar xvf openshift-install-linux.tar.gz
```

5. 从 Red Hat OpenShift Cluster Manager 下载安装 pull secret 。通过此 pull secret，您可以进行所含授权机构提供的服务的身份验证，这些服务包括为 OpenShift Container Platform 组件提供容器镜像的 Quay.io。

2.7.6. 创建安装配置文件

您可以自定义在 Amazon Web Services (AWS) 上安装的 OpenShift Container Platform 集群。

先决条件

- 获取 OpenShift Container Platform 安装程序以及集群的 pull secret。

流程

1. 创建 **install-config.yaml** 文件。
   a. 更改到包含安装程序的目录，再运行以下命令：

```
$ ./openshift-install create install-config --dir <installation_directory> 1
```

1 对于 **<installation_directory>**，请指定用于保存安装程序所创建的文件的目录名称。

重要

指定一个空目录。一些安装信息，如 bootstrap X.509 证书，有较短的过期间隔，因此不要重复使用安装目录。如果要重复使用另一个集群安装中的个别文件，可以将其复制到您的目录中。但是，一些安装数据的文件名可能会在发行版本之间有所改变。从 OpenShift Container Platform 老版本中复制安装文件时要格外小心。

b. 在提示符处，提供您的云的配置详情：

i. 可选：选择用来访问集群机器的 SSH 密钥。

注意

对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 **ssh-agent** 进程使用的 SSH 密钥。

ii. 选择 **AWS** 作为目标平台。

iii. 如果计算机上没有保存 Amazon Web Services (AWS) 配置集，请为您配置用于运行安装程序的用户输入 AWS 访问密钥 ID 和 Secret 访问密钥。

iv. 选择要将集群部署到的 AWS 区域。

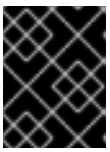
v. 选择您为集群配置的 Route 53 服务的基域。

vi. 为集群输入一个描述性名称。

vii. 粘贴 Red Hat OpenShift Cluster Manager 中的 pull secret 。

2. 修改 **install-config.yaml** 文件。您可以在**安装配置参数**部分中找到有关可用参数的更多信息。

3. 备份 **install-config.yaml** 文件，以便用于安装多个集群。



重要

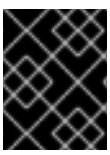
install-config.yaml 文件会在安装过程中消耗掉。如果要重复使用此文件，必须现在备份。

2.7.6.1. 安装配置参数

在部署 OpenShift Container Platform 集群前，您可以提供参数值，以描述托管集群的云平台的帐户并选择性地自定义集群平台。在创建 **install-config.yaml** 安装配置文件时，您可以通过命令行来提供所需的参数的值。如果要自定义集群，可以修改 **install-config.yaml** 文件来提供关于平台的更多信息。

注意

安装之后，您无法修改 **install-config.yaml** 文件中的这些参数。



重要

openshift-install 命令不验证参数的字段名称。如果指定了不正确的名称，则不会创建相关的文件或对象，且不会报告错误。确保所有指定的参数的字段名称都正确。

2.7.6.1.1. 所需的配置参数

下表描述了所需的安装配置参数：

表 2.18. 所需的参数

参数	描述	值
apiVersion	**install-config.yaml** 内容的 API 版本。当前版本是 **v1**。安装程序还可能支持旧的 API 版本。	字符串
baseDomain	云供应商的基域。此基础域用于创建到 OpenShift Container Platform 集群组件的路由。集群的完整 DNS 名称是 **baseDomain** 和 **metadata.name** 参数值的组合，其格式为 **<metadata.name>.<baseDomain>**。	完全限定域名或子域名，如 **example.com**。
metadata	Kubernetes 资源 **ObjectMeta**，其中只消耗 **name** 参数。	对象
metadata.name	集群的名称。集群的 DNS 记录是 **{{.metadata.name}}.{{.baseDomain}}** 的子域。	小写字母,连字符(-)和句点(.)的字符串，如 **dev**。
platform	执行安装的具体平台配置：**aws**、**baremetal**、**azure**、**openstack**、**ovirt**、**vsphere**。有关 **platform.<platform>** 参数的额外信息，请参考下表来了解您的具体平台。	对象

pullSecret	从 Red Hat OpenShift Cluster Manager 获取 pull secret，验证从 Quay.io 等服务中下载 OpenShift Container Platform 组件的容器镜像。	{
		"auths":{
		"cloud.openshift.com":{
		"auth":"b3Blb=",
		"email":"you@example.com"
		},
		"quay.io":{
		"auth":"b3Blb=",
		"email":"you@example.com"
		}
		}
		}

2.7.6.1.2. 网络配置参数

您可以根据现有网络基础架构的要求自定义安装配置。例如，您可以扩展集群网络的 IP 地址块，或者提供不同于默认值的不同 IP 地址块。

只支持 IPv4 地址。

表 2.19. 网络参数

参数	描述	值
networking	集群网络的配置。	对象 **注意** 您不能在安装后修改 **networking** 对象指定的参数。
networking.networkType	要安装的集群网络供应商 Container Network Interface（CNI）插件。	**OpenShiftSDN** 或 **OVNKubernetes**。默认值为 **OpenShiftSDN**。

networking.clusterNetwork	pod 的 IP 地址块。
	
	默认值为 **10.128.0.0/14**，主机前缀为 **/23**。
	
	如果您指定多个 IP 地址块，则块不得互相重叠。
		一个对象数组。例如：
		
		``
		networking:
		  clusterNetwork:
		  - cidr: 10.128.0.0/14
		    hostPrefix: 23
		``

networking.clusterNetwork.cidr	使用 **networking.clusterNetwork** 时需要此项。IP 地址块。 一个 IPv4 网络。	使用 CIDR 形式的 IP 地址块。IPv4 块的前缀长度介于 **0** 到 **32** 之间。
networking.clusterNetwork.hostPrefix	分配给每个单独节点的子网前缀长度。例如，如果 **hostPrefix** 设为 **23**，则每个节点从所给的 **cidr** 中分配一个 **/23** 子网。**hostPrefix** 值 **23** 提供 510（2^(32 - 23)- 2）个 pod IP 地址。	子网前缀。 默认值为 **23**。

networking.serviceNetwork	服务的 IP 地址块。默认值为 **172.30.0.0/16**。
	
	OpenShift SDN 和 OVN-Kubernetes 网络供应商只支持服务网络的一个 IP 地址块。
		CIDR 格式具有 IP 地址块的数组。例如：
		
		``
		networking:
		  serviceNetwork:
		  - 172.30.0.0/16
		``

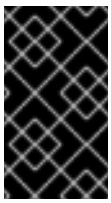
参数	描述	值
networking.machineNetwork	机器的 IP 地址块。 如果您指定多个 IP 地址块，则块不得互相重叠。	一个对象数组。例如： networking: machineNetwork: - cidr: 10.0.0.0/16
networking.machineNetwork.cidr	使用 **networking.machineNetwork** 时需要。IP 地址块。libvirt 以外的所有平台的默认值为 **10.0.0.0/16**。对于 libvirt，默认值为 **192.168.126.0/24**。	CIDR 表示法中的 IP 网络块。 例如： **10.0.0.0/16**。 **注意** 将 **networking.machineNetwork** 设置为与首选 NIC 所在的 CIDR 匹配。

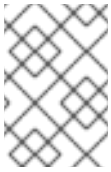
2.7.6.1.3. 可选配置参数

下表描述了可选安装配置参数：

表 2.20. 可选参数

参数	描述	值
additionalTrustBundle	添加到节点可信证书存储中的 PEM 编码 X.509 证书捆绑包。配置了代理时，也可以使用这个信任捆绑包。	字符串
compute	组成计算节点的机器的配置。	machine-pool 对象的数组。详情请查看以下"Machine-pool"表。
compute.architecture	决定池中机器的指令集合架构。目前不支持异构集群，因此所有池都必须指定相同的架构。有效值为 **amd64** （默认值）。	字符串

参数	描述	值
compute.hyperthreading	是否在计算机器上启用或禁用并发多线程或**超线程**。默认情况下，启用并发多线程以提高机器内核的性能。  **重要** 如果禁用并发多线程，请确保在容量规划时考虑到机器性能可能会显著降低的问题。	**Enabled** 或 **Disabled**
compute.name	使用 **compute** 时需要此值。机器池的名称。	**worker**
compute.platform	使用 **compute** 时需要此值。使用此参数指定托管 worker 机器的云供应商。此参数值必须与 **controlPlane.platform** 参数值匹配。	**aws**、**azure**、**gcp**、**openstack**、**ovirt**、**vsphere** 或 **{}**
compute.replicas	要置备的计算机器数量，也称为 worker 机器。	大于或等于 **2** 的正整数。默认值为 **3**。
controlPlane	组成 control plane 的机器的配置。	**MachinePool** 对象的数组。详情请查看以下"Machine-pool"表。
controlPlane.architecture	决定池中机器的指令集合架构。目前不支持异构集群，因此所有池都必须指定相同的架构。有效值为 **amd64** （默认值）。	字符串
controlPlane.hyperthreading	是否在 control plane 机器上启用或禁用并发多线程或**超线程**。默认情况下，启用并发多线程以提高机器内核的性能。  **重要** 如果禁用并发多线程，请确保在容量规划时考虑到机器性能可能会显著降低的问题。	**Enabled** 或 **Disabled**
controlPlane.name	使用 **controlPlane** 时需要。机器池的名称。	**master**

参数	描述	值
controlPlane.platform	使用 **controlPlane** 时需要。使用此参数指定托管 control plane 机器的云供应商。此参数值必须与 **compute.platform** 参数值匹配。	**aws**、**azure**、**gcp**、**openstack**、**ovirt**、**vsphere** 或 **{}**
controlPlane.replicas	要置备的 control plane 机器数量。	唯一支持的值是 **3**，它是默认值。
credentialsMode	Cloud Credential Operator（CCO）模式。如果没有指定任何模式，CCO 会动态地尝试决定提供的凭证的功能，在支持多个模式的平台上使用 mint 模式。  **注意** 不是所有 CCO 模式都支持所有云供应商。如需有关 CCO 模式的更多信息，请参阅*Red Hat Operator 参考指南*内容中的 *Cloud Credential Operator* 条目。	**Mint**、**Passthrough**、**Manual** 或空字符串(**""**)。
fips	启用或禁用 FIPS 模式。默认为 **false**（禁用）。如果启用了 FIPS 模式，运行 OpenShift Container Platform 的 Red Hat Enterprise Linux CoreOS (RHCOS) 机器会绕过默认的 Kubernetes 加密套件，并使用由 RHCOS 提供的加密模块。  **重要** 只有在 **x86_64** 架构中的 OpenShift Container Platform 部署支持 FIPS 验证的/Modules in Process 加密库。  **注意** 如果使用 Azure File 存储，则无法启用 FIPS 模式。	**false** 或 **true**

参数	描述	值
imageContentSources	release-image 内容的源和仓库。	对象数组。包括一个 **source** 以及可选的 **mirrors**，如下表所示。
imageContentSources.source	使用 **imageContentSources** 时需要。指定用户在镜像拉取规格中引用的仓库。	字符串
imageContentSources.mirrors	指定可能还包含同一镜像的一个或多个仓库。	字符串数组
publish	如何发布或公开集群的面向用户的端点，如 Kubernetes API、OpenShift 路由。	**Internal** 或 **External**。把 **publish** 设置为 **Internal** 以部署一个私有集群，它不能被互联网访问。默认值为 **External**。
sshKey	用于验证集群机器访问的 SSH 密钥或密钥。 **注意** 对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 **ssh-agent** 进程使用的 SSH 密钥。	一个或多个密钥。例如： sshKey: <key1> <key2> <key3>

2.7.6.1.4. 可选的 AWS 配置参数

下表描述了可选的 AWS 配置参数：

表 2.21. 可选的 AWS 参数

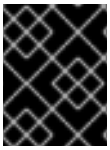
参数	描述	值
compute.platform.aws.amiID	用于为集群引导计算机器的 AWS AMI。对于需要自定义 RHCOS AMI 的区域来说，这是必需的。	属于集合 AWS 区域的任何已发布或自定义 RHCOS AMI。
compute.platform.aws.rootVolume.iops	为根卷保留的每秒输入/输出操作 (IOPS) 数。	整数，如 **4000**。

参数	描述	值
compute.platform.aws.rootVolume.size	以 GiB 为单位的根卷大小。	整数，如 **500**。
compute.platform.aws.rootVolume.type	根卷的类型。	有效的 AWS EBS 卷类型，如 **io1**。
compute.platform.aws.rootVolume.kmsKeyARN	KMS 密钥的 Amazon 资源名称（密钥 ARN）。这是使用特定 KMS 密钥加密 worker 节点的操作系统卷。	有效的 密钥 ID 或密钥 ARN。
compute.platform.aws.type	计算机器的 EC2 实例类型。	有效的 AWS 实例类型，如 **c5.9xlarge**。
compute.platform.aws.zones	安装程序在其中为计算机机器池创建机器的可用区。如果您提供自己的 VPC，则必须在那个可用域中提供一个子网。	有效 AWS 可用区的列表，如 **us-east-1c**，以 YAML 序列表示。
compute.aws.region	安装程序在其中创建计算资源的 AWS 区域。	任何有效的 AWS 区域，如 **us-east-1**。
controlPlane.platform.aws.amiID	用于为集群引导 control plane 机器的 AWS AMI。对于需要自定义 RHCOS AMI 的区域来说，这是必需的。	属于集合 AWS 区域的任何已发布或自定义 RHCOS AMI。
controlPlane.platform.aws.rootVolume.kmsKeyARN	KMS 密钥的 Amazon 资源名称（密钥 ARN）。这需要使用特定的 KMS 密钥加密 control plane 节点的操作系统卷。	有效的 密钥 ID 和密钥 ARN。
controlPlane.platform.aws.type	control plane 机器的 EC2 实例类型。	有效的 AWS 实例类型，如 **c5.9xlarge**。
controlPlane.platform.aws.zones	安装程序在其中为 control plane 机器池创建机器的可用区。	有效 AWS 可用区的列表，如 **us-east-1c**，以 YAML 序列表示。
controlPlane.aws.region	安装程序在其中创建 control plane 资源的 AWS 区域。	有效的 AWS 区域，如 **us-east-1**。
platform.aws.amiID	用于为集群引导所有机器的 AWS AMI。如果设置，AMI 必须属于与集群相同的区域。对于需要自定义 RHCOS AMI 的区域来说，这是必需的。	属于集合 AWS 区域的任何已发布或自定义 RHCOS AMI。

参数	描述	值
platform.aws.serviceEndpoints.name	AWS 服务端点名称。只有在必须使用替代 AWS 端点（如 FIPS）时，才需要自定义端点。可以为 EC2、S3、IAM、Elastic Load Balancing、Tagging、Route 53 和 STS AWS 服务指定自定义 API 端点。	有效的 AWS 服务端点名称。
platform.aws.serviceEndpoints.url	AWS 服务端点 URL。URL 必须使用 **https** 协议，主机必须信任该证书。	有效的 AWS 服务端点 URL。
platform.aws.userTags	键与值的映射，安装程序将其作为标签添加到它所创建的所有资源。	任何有效的 YAML 映射，如 **<key>: <value>** 格式的键值对。如需有关 AWS 标签的更多信息，请参阅 AWS 文档中的标记您的 Amazon EC2 资源。
platform.aws.subnets	如果您提供 VPC，而不是让安装程序为您创建 VPC，请指定要使用的集群子网。子网必须是您指定的同一 **machineNetwork[].cidr** 范围的一部分。对于标准集群，为每个可用区指定一个公共和私有子网。对于私有集群，为每个可用区指定一个私有子网。	有效的子网 ID。

2.7.6.2. AWS 的自定义 install-config.yaml 文件示例

您可以自定义 **install-config.yaml** 文件，以指定有关 OpenShift Container Platform 集群平台的更多信息，或修改所需参数的值。



重要

此示例 YAML 文件仅供参考。您必须使用安装程序来获取 **install-config.yaml** 文件，并且修改该文件。

```
apiVersion: v1
baseDomain: example.com (1)
credentialsMode: Mint (2)
controlPlane: (3) (4)
  hyperthreading: Enabled (5)
  name: master
  platform:
    aws:
      zones:
```

```
      - us-west-2a
      - us-west-2b
      rootVolume:
        iops: 4000
        size: 500
        type: io1 6
      type: m5.xlarge
  replicas: 3
compute: 7
- hyperthreading: Enabled 8
  name: worker
  platform:
    aws:
      rootVolume:
        iops: 2000
        size: 500
        type: io1 9
      type: c5.4xlarge
      zones:
      - us-west-2c
  replicas: 3
metadata:
  name: test-cluster 10
networking:
  clusterNetwork:
  - cidr: 10.128.0.0/14
    hostPrefix: 23
  machineNetwork:
  - cidr: 10.0.0.0/16
  networkType: OpenShiftSDN
  serviceNetwork:
  - 172.30.0.0/16
platform:
  aws:
    region: us-west-2 11
    userTags:
      adminContact: jdoe
      costCenter: 7536
    subnets: 12
    - subnet-1
    - subnet-2
    - subnet-3
    amiID: ami-96c6f8f7 13
    serviceEndpoints: 14
      - name: ec2
        url: https://vpce-id.ec2.us-west-2.vpce.amazonaws.com
    hostedZone: Z3URY6TWQ91KVV 15
fips: false 16
sshKey: ssh-ed25519 AAAA... 17
pullSecret: '{"auths": ...}' 18
```

1 10 11 18 必需。安装程序会提示您输入这个值。

2 可选：添加此参数来强制 Cloud Credential Operator（CCO）使用指定的模式，而不是让 CCO 动

3 7 如果没有提供这些参数和值，安装程序会提供默认值。

4 **controlPlane** 部分是一个单映射，但 **compute** 部分是一系列映射。为满足不同数据结构的要求，**compute** 部分的第一行必须以连字符 **-** 开头，**controlPlane** 部分的第一行则不可以连字符开头。只使用一个 control plane 池。

5 8 是否要启用或禁用并发多线程或**超线程**。默认情况下，启用并发多线程以提高机器内核的性能。您可以通过将参数值设为 **Disabled** 来禁用。如果您在某些集群机器上禁用并发多线程，则必须在所有集群机器上禁用。

重要

如果禁用并发多线程，请确保在容量规划时考虑到机器性能可能会显著降低的问题。如果您对机器禁用并发多线程，请使用较大的实例类型，如 **m4.2xlarge** 或 **m5.2xlarge**。

6 9 要为 etcd 配置更快的存储，特别是对于较大的集群，请将存储类型设置为 **io1**，并将 **iops** 设为 **2000**。

12 如果您提供自己的 VPC，为集群使用的每个可用区指定子网。

13 用于为集群引导机器的 AMI ID。如果设置，AMI 必须属于与集群相同的区域。

14 AWS 服务端点。在安装到未知 AWS 区域时，需要自定义端点。端点 URL 必须使用 **https** 协议，主机必须信任该证书。

15 您现有 Route 53 私有托管区的 ID。提供现有的托管区需要您提供自己的 VPC，托管区已在安装集群前与 VPC 关联。如果未定义，安装程序会创建一个新的托管区。

16 是否启用或禁用 FIPS 模式。默认情况下不启用 FIPS 模式。如果启用了 FIPS 模式，运行 OpenShift Container Platform 的 Red Hat Enterprise Linux CoreOS (RHCOS) 机器会绕过默认的 Kubernetes 加密套件，并使用由 RHCOS 提供的加密模块。

重要

只有在 **x86_64** 架构中的 OpenShift Container Platform 部署支持 FIPS 验证的/Modules in Process 加密库。

17 您可以选择提供您用来访问集群中机器的 **sshKey** 值。

注意

对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 **ssh-agent** 进程使用的 SSH 密钥。

2.7.6.3. 在安装过程中配置集群范围代理

生产环境可能会拒绝直接访问互联网，而是提供 HTTP 或 HTTPS 代理。您可以通过在 **install-config.yaml** 文件中配置代理设置，将新的 OpenShift Container Platform 集群配置为使用代理。

先决条件

- 您有一个现有的 **install-config.yaml** 文件。

- 您检查了集群需要访问的站点，并决定是否需要绕过代理。默认情况下代理所有集群出口流量，包括对托管云供应商 API 的调用。您需要将站点添加到 **Proxy** 对象的 **spec.noProxy** 字段来绕过代理。

注意

Proxy 对象 **status.noProxy** 字段使用安装配置中的 **networking.machineNetwork[].cidr**、**networking.clusterNetwork[].cidr** 和 **networking.serviceNetwork[]** 字段的值填充。

对于在 Amazon Web Services(AWS)、Google Cloud Platform(GCP)、Microsoft Azure 和 Red Hat OpenStack Platform(RHOSP)上安装, **Proxy** 对象 **status.noProxy** 字段也会使用实例元数据端点填充(**169.254.169.254**)。

- 如果您的集群位于 AWS 上，请将 **ec2.<region>.amazonaws.com**、**elasticloadbalancing.<region>.amazonaws.com** 和 **s3.<region>.amazonaws.com** 端点添加到 VPC 端点。需要这些端点才能完成节点到 AWS EC2 API 的请求。由于代理在容器级别而不是节点级别工作，因此您必须通过 AWS 专用网络将这些请求路由到 AWS EC2 API。在代理服务器中的允许列表中添加 EC2 API 的公共 IP 地址是不够的。

流程

1. 编辑 **install-config.yaml** 文件并添加代理设置。例如：

```
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: example.com 3
additionalTrustBundle: | 4
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
...
```

1. 用于创建集群外 HTTP 连接的代理 URL。URL 必须是 **http**。
2. 用于创建集群外 HTTPS 连接的代理 URL。
3. 要排除在代理中的目标域名、IP 地址或其他网络 CIDR 的逗号分隔列表。在域前面加 **.** 来仅匹配子域。例如： **.y.com** 匹配 **x.y.com**，但不匹配 **y.com**。使用 * 绕过所有目的地的代理。
4. 如果提供，安装程序会在 **openshift-config** 命名空间中生成名为 **user-ca- bundle** 的配置映射来保存额外的 CA 证书。如果您提供 **additionalTrustBundle** 和至少一个代理设置，则 **Proxy** 对象会被配置为引用 **trustedCA** 字段中的 **user-ca-bundle** 配置映射。然后，Cluster Network Operator 会创建一个 **trusted-ca-bundle** 配置映射，该配置映射将为 **trustedCA** 参数指定的内容与 RHCOS 信任捆绑包合并。 **additionalTrustBundle** 字段是必需的，除非代理的身份证书由来自 RHCOS 信任捆绑包的颁发机构签名。

注意

安装程序不支持代理的 **readinessEndpoints** 字段。

2. 保存该文件，并在安装 OpenShift Container Platform 时引用。

安装程序会创建一个名为 **cluster** 的集群范围代理，该代理使用提供的 **install-config.yaml** 文件中的代理设置。如果没有提供代理设置，仍然会创建一个 **cluster Proxy** 对象，但它会有一个空 **spec**。

注意

只支持名为 **cluster** 的 **Proxy** 对象，且无法创建额外的代理。

2.7.7. 部署集群

您可以在兼容云平台中安装 OpenShift Container Platform。

重要

安装程序的 **create cluster** 命令只能在初始安装过程中运行一次。

先决条件

- 配置托管集群的云平台的帐户。
- 获取 OpenShift Container Platform 安装程序以及集群的 pull secret。

流程

1. 更改为包含安装程序的目录并初始化集群部署：

```
$ ./openshift-install create cluster --dir <installation_directory> \ 1
    --log-level=info 2
```

1 对于 **<installation_directory>**，请指定自定义 **./install-config.yaml** 文件的位置。

2 要查看不同的安装详情，请指定 **warn**、**debug** 或 **error**，而不要指定 **info**。

注意

如果您在主机上配置的云供应商帐户没有足够的权限来部署集群，安装过程将会停止，并且显示缺少的权限。

集群部署完成后，终端会显示访问集群的信息，包括指向其 Web 控制台的链接和 **kubeadmin** 用户的凭证。

输出示例

```
...
INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run 'export
KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'
```

```
INFO Access the OpenShift web-console here: https://console-openshift-
console.apps.mycluster.example.com
INFO Login to the console with user: "kubeadmin", and password: "4vYBz-Ee6gm-ymBZj-
Wt5AL"
INFO Time elapsed: 36m22s
```

注意

当安装成功时，集群访问和凭证信息还会输出到 **<installation_directory>/.openshift_install.log**。

重要

- 安装程序生成的 Ignition 配置文件包含在 24 小时后过期的证书，然后在过期时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外情况是，您需要手动批准待处理的 **node-bootstrapper** 证书签名请求（CSR）来恢复 kubelet 证书。如需更多信息，请参阅*从过期的 control plane 证书中恢复*的文档。
- 建议您在生成 12 小时后使用 Ignition 配置文件，因为集群安装后 24 小时证书从 16 小时轮转至 22 小时。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中运行证书更新时避免安装失败。

重要

您不得删除安装程序或安装程序所创建的文件。需要这两者才能删除集群。

2. 可选：从您用来安装集群的 IAM 帐户删除或禁用 **AdministratorAccess** 策略。

注意

只有在安装过程中才需要 **AdministratorAccess** 策略提供的升级权限。

2.7.8. 通过下载二进制文件安装 OpenShift CLI

您需要安装 CLI（**oc**） 来使用命令行界面与 OpenShift Container Platform 进行交互。您可在 Linux、Windows 或 macOS 上安装 **oc**。

重要

如果安装了旧版本的 **oc**，则无法使用 OpenShift Container Platform 4.6 中的所有命令。下载并安装新版本的 **oc**。

2.7.8.1. 在 Linux 上安装 OpenShift CLI

您可以按照以下流程在 Linux 上安装 OpenShift CLI（**oc**）二进制文件。

流程

1. 进入到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 在 **Version** 下拉菜单中选择相应的版本。

3. 单击 **OpenShift v4.6 Linux 客户端** 条目旁边的 **Download Now**，再保存文件。
4. 解包存档：

```
$ tar xvzf <file>
```

5. 把 **oc** 二进制代码放到 **PATH** 中的目录中。
执行以下命令可以查看当前的 **PATH** 设置：

```
$ echo $PATH
```

安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
$ oc <command>
```

2.7.8.2. 在 Windows 上安装 OpenShift CLI

您可以按照以下流程在 Windows 上安装 OpenShift CLI（**oc**）二进制代码。

流程

1. 进入到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 在 **Version** 下拉菜单中选择相应的版本。
3. 单击 **OpenShift v4.6 Windows 客户端**条目旁边的 **Download Now**，再保存文件。
4. 使用 ZIP 程序解压存档。
5. 把 **oc** 二进制代码放到 **PATH** 中的目录中。
要查看您的 **PATH**，请打开命令提示窗口并执行以下命令：

```
C:\> path
```

安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
C:\> oc <command>
```

2.7.8.3. 在 macOS 上安装 OpenShift CLI

您可以按照以下流程在 macOS 上安装 OpenShift CLI（**oc**）二进制代码。

流程

1. 进入到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 在 **Version** 下拉菜单中选择相应的版本。
3. 单击 **OpenShift v4.6 MacOSX 客户端**条目旁边的 **Download Now**，再保存文件。
4. 解包和解压存档。
5. 将 **oc** 二进制文件移到 PATH 的目录中。

要查看您的 **PATH**，打开一个终端窗口并执行以下命令：

```
$ echo $PATH
```

安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
$ oc <command>
```

2.7.9. 使用 CLI 登录到集群

您可以通过导出集群 **kubeconfig** 文件，以默认系统用户身份登录集群。**kubeconfig** 文件包含关于集群的信息，供 CLI 用于将客户端连接到正确集群和 API 服务器。该文件特只适用于一个特定的集群，在 OpenShift Container Platform 安装过程中创建。

先决条件

- 已部署了 OpenShift Container Platform 集群。
- 已安装 **oc** CLI。

流程

1. 导出 **kubeadmin** 凭证：

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

2. 使用导出的配置，验证能否成功运行 **oc** 命令：

```
$ oc whoami
```

输出示例

```
system:admin
```

2.7.10. 使用 Web 控制台登录到集群

kubeadmin 用户默认在 OpenShift Container Platform 安装后存在。您可以使用 OpenShift Container Platform Web 控制台以 **kubeadmin** 用户身份登录集群。

先决条件

- 有访问安装主机的访问权限。
- 您完成了集群安装，所有集群 Operator 都可用。

流程

1. 从安装主机上的 **kubeadmin -password** 文件中获取 kubeadmin 用户的密码：

```
$ cat <installation_directory>/auth/kubeadmin-password
```

注意

另外，您还可以从安装主机上的 **<installation_directory>/.openshift_install.log** 日志文件获取 **kubeadmin** 密码。

2. 列出 OpenShift Container Platform Web 控制台路由：

```
$ oc get routes -n openshift-console | grep 'console-openshift'
```

注意

另外，您还可以从安装主机上的 **<installation_directory>/.openshift_install.log** 日志 文件获取 OpenShift Container Platform 路由。

输出示例

```
console     console-openshift-console.apps.<cluster_name>.<base_domain>           console
https   reencrypt/Redirect   None
```

3. 在 Web 浏览器中导航到上一命令输出中包括的路由，以 **kubeadmin** 用户身份登录。

其他资源

- 如需有关访问和了解 OpenShift Container Platform Web 控制台的更多信息，请参阅访问 Web 控制台。

2.7.11. OpenShift Container Platform 的 Telemetry 访问

在 OpenShift Container Platform 4.6 中，默认运行的 Telemetry 服务提供有关集群健康状况和成功更新的指标，需要访问互联网。如果您的集群连接到互联网，Telemetry 会自动运行，而且集群会注册到 OpenShift Cluster Manager。

确认 OpenShift Cluster Manager 清单正确后，可以由 Telemetry 自动维护，也可以使用 OpenShift Cluster Manager 手动维护，使用订阅监控来跟踪帐户或多集群级别的 OpenShift Container Platform 订阅。

其他资源

- 有关 Telemetry 服务的更多信息，请参阅关于远程健康监控。

2.7.12. 后续步骤

- 验证安装。
- 自定义集群。
- 如果需要，您可以选择不使用远程健康报告。
- 如果需要，您可以删除云供应商凭证。

2.8. 在 AWS 上安装私有集群

在 OpenShift Container Platform 版本 4.6 中，您可以在 Amazon Web Services（AWS）上将私有集群安装到现有的 VPC 中。安装程序会置备所需基础架构的其余部分，您可以进一步定制这些基础架构。要自定义安装，请在安装集群前修改 **install-config.yaml** 文件中的参数。

2.8.1. 先决条件

- 查看有关 OpenShift Container Platform 安装和更新流程的详细信息。
- 配置 AWS 帐户以托管集群。

重要

如果您的计算机上存储有 AWS 配置集，则不要在使用多因素验证设备的同时使用您生成的临时会话令牌。在集群的整个生命周期中，集群会持续使用您的当前 AWS 凭证来创建 AWS 资源，因此您必须使用长期凭证。要生成适当的密钥，请参阅 AWS 文档中的管理 IAM 用户的访问密钥。您可在运行安装程序时提供密钥。

- 如果使用防火墙，则必须将其配置为允许集群需要访问的站点。
- 如果不允许系统管理身份和访问管理（IAM），集群管理员可以 手动创建和维护 IAM 凭证。手动模式也可以用于云 IAM API 无法访问的环境中。

2.8.2. 私有集群

您可以部署不公开外部端点的私有 OpenShift Container Platform 集群。私有集群只能从内部网络访问，且无法在互联网中看到。

默认情况下，OpenShift Container Platform 被置备为使用可公开访问的 DNS 和端点。私有集群在部署集群时将 DNS 、Ingress Controller 和 API 服务器设置为私有。这意味着，集群资源只能从您的内部网络访问，且不能在互联网中看到。

要部署私有集群，您必须使用符合您的要求的现有网络。您的集群资源可能会在网络中的其他集群间共享。

另外，您必须从可访问您置备的云的 API 服务、您置备的网络上的主机以及可以连接到互联网来获取安装介质的机器上部署私有集群。您可以使用符合这些访问要求的机器，并按照您的公司规定进行操作。例如，该机器可以是云网络中的堡垒主机，也可以是可通过 VPN 访问网络的机器。

2.8.2.1. AWS 中的私有集群

要在 Amazon Web Services (AWS) 上创建私有集群，您必须提供一个现有的私有 VPC 和子网来托管集群。安装程序还必须能够解析集群所需的 DNS 记录。安装程序将 Ingress Operator 和 API 服务器配置为只可以从私有网络访问。

集群仍然需要访问互联网来访问 AWS API。

安装私有集群时不需要或创建以下项目：

- 公共子网
- 支持公共入口的公共负载均衡器
- 与集群的 **baseDomain** 匹配的公共 Route 53 区域

安装程序会使用您指定的 **baseDomain** 来创建专用的 Route 53 区域以及集群所需的记录。集群被配置，以便 Operator 不会为集群创建公共记录，且所有集群机器都放置在您指定的私有子网中。

2.8.2.1.1. 限制：

为私有集群添加公共功能的能力有限。

- 在安装后，您无法在不进行额外操作的情况下公开 Kubernetes API 端点。这些额外的操作包括为使用中的每个可用区在 VPC 中创建公共子网，创建公共负载均衡器，以及配置 control plane 安全组以便 6443 端口（Kubernetes API 端口）可以接受来自于互联网的网络流量。
- 如果使用公共服务类型负载均衡器，您必须在每个可用区中为公共子网添加 **kubernetes.io/cluster/<cluster-infra-id>: shared** 标签，以便 AWS 可使用它们来创建公共负载均衡器。

2.8.3. 关于使用自定义 VPC

在 OpenShift Container Platform 4.6 中，您可以在 Amazon Web Services（AWS）的现有 Amazon Virtual Private Cloud（VPC）中将集群部署到现有子网中。通过将 OpenShift Container Platform 部署到现有的 AWS VPC 中，您可能会避开新帐户中的限制，或者更容易地利用公司所设置的操作限制。如果您无法获得您自己创建 VPC 所需的基础架构创建权限，请使用这个安装选项。

因为安装程序无法了解您现有子网中还有哪些其他组件，所以无法选择子网 CIDR 。您必须为安装集群的子网配置网络。

2.8.3.1. 使用 VPC 的要求

安装程序不再创建以下组件：

- 互联网网关
- NAT 网关
- 子网
- 路由表
- VPCs
- VPC DHCP 选项
- VPC 端点

注意

安装程序要求您使用由云提供的 DNS 服务器。不支持使用自定义 DNS 服务器，并导致安装失败。

如果您使用自定义 VPC，您必须为安装程序和集群正确配置它及其子网。有关创建和管理 AWS VPC VPC 的更多信息，请参阅 AWS 文档中的 Amazon VPC 控制台向导配置 以及处理 VPC 和子网。

安装程序无法：

- 子类网络范围供群集使用。

- 设置子网的路由表。
- 设置 VPC 选项，如 DHCP。

您必须在安装集群前完成这些任务。有关在 AWS VPC 中配置网络的更多信息，请参阅 VPC 的 VPC 网络组件 和路由表。

您的 VPC 必须满足以下特征：

- VPC 不能使用 **kubernetes.io/cluster/.*: owned** 标签。
  安装程序会修改子网以添加 **kubernetes.io/cluster/.*: shared** 标签，因此您的子网必须至少有一个可用的空闲标签插槽。请参阅 AWS 文档中的 标签限制 部分，以确认安装程序可以为您指定的每个子网添加标签。
- 您必须在 VPC 中启用 **enable DnsSupport 和 enableDnsHostnames** 属性，以便集群可以使用附加到 VPC 的 Route 53 区域来解析集群内部 DNS 记录。请参阅 AWS 文档中的您的 VPC 中的 DNS 支持部分 。
  如果您希望使用自己的 Route 53 托管私有区，则必须在安装集群前将现有托管区与 VPC 关联。您可以使用 **install-config.yaml** 文件中的 **platform.aws.hostedZone** 字段定义托管区。
- 如果您使用具有公共访问权限的集群，您必须为每个集群使用的可用区创建一个公共和私有子网。每个可用区不能包含多于一个的公共子网和私有子网。

如果您在断开连接的环境中工作，您将无法访问 EC2 和 ELB 端点的公共 IP 地址。要解决这个问题，您必须创建一个 VPC 端点，并将其附加到集群使用的子网。端点应命名如下：

- **ec2.<region>.amazonaws.com**
- **elasticloadbalancing.<region>.amazonaws.com**
- **s3.<region>.amazonaws.com**

所需的 VPC 组件

您必须提供合适的 VPC 和子网，以便与您的机器通信。

组件	AWS 类型	描述
VPC	• **AWS::EC2::VPC** • **AWS::EC2::VPCEndpoint**	您必须提供一个公共 VPC 供集群使用。VPC 使用引用每个子网的路由表的端点，以改进与托管在 S3 中的 registry 的通信。
公共子网	• **AWS::EC2::Subnet** • **AWS::EC2::SubnetNetworkAclAssociation**	您的 VPC 必须有 1 到 3 个可用区的公共子网，并将其与适当的入口规则关联。

<table>
<tr><th>组件</th><th>AWS 类型</th><th colspan="2">描述</th></tr>
<tr><td>互联网网关</td><td>AWS::EC2::InternetGateway
AWS::EC2::VPCGatewayAttachment
AWS::EC2::RouteTable
AWS::EC2::Route
AWS::EC2::SubnetRouteTableAssociation
AWS::EC2::NatGateway
AWS::EC2::EIP</td><td colspan="2">您必须有一个公共互联网网关，以及附加到 VPC 的公共路由。在提供的模板中，每个公共子网都有一个具有 EIP 地址的 NAT 网关。这些 NAT 网关允许集群资源（如专用子网实例）访问互联网，而有些受限网络或代理场景则不需要它们。</td></tr>
<tr><td>网络访问控制</td><td>AWS::EC2::NetworkAcl
AWS::EC2::NetworkAclEntry</td><td colspan="2">您必须允许 VPC 访问下列端口：</td></tr>
<tr><td></td><td></td><td>端口</td><td>原因</td></tr>
<tr><td></td><td></td><td>80</td><td>入站 HTTP 流量</td></tr>
<tr><td></td><td></td><td>443</td><td>入站 HTTPS 流量</td></tr>
<tr><td></td><td></td><td>22</td><td>入站 SSH 流量</td></tr>
<tr><td></td><td></td><td>1024 - 65535</td><td>入站临时流量</td></tr>
<tr><td></td><td></td><td>0 - 65535</td><td>出站临时流量</td></tr>
<tr><td>专用子网</td><td>AWS::EC2::Subnet
AWS::EC2::RouteTable
AWS::EC2::SubnetRouteTableAssociation</td><td colspan="2">您的 VPC 可以具有私有子网。提供的 CloudFormation 模板可为 1 到 3 个可用区创建专用子网。如果您使用专用子网，必须为其提供适当的路由和表。</td></tr>
</table>

2.8.3.2. VPC 验证

要确保您提供的子网适合您的环境，安装程序会确认以下信息：

- 您指定的所有子网都存在。
- 您提供了私有子网。
- 子网 CIDR 属于您指定的机器 CIDR。
- 您为每个可用区提供子网。每个可用区不包含多于一个的公共子网和私有子网。如果您使用私有集群，为每个可用区只提供一个私有子网。否则，为每个可用区提供一个公共和私有子网。

- 您可以为每个私有子网可用区提供一个公共子网。机器不会在没有为其提供私有子网的可用区中置备。

如果您销毁使用现有 VPC 的集群，VPC 不会被删除。从 VPC 中删除 OpenShift Container Platform 集群时，**kubernetes.io/cluster/.*: shared** 标签会从使用它的子网中删除。

2.8.3.3. 权限划分

从 OpenShift Container Platform 4.3 开始，您不需要安装程序置备的基础架构集群部署所需的所有权限。这与您所在机构可能已有的权限划分类似：不同的个人可以在您的云中创建不同的资源。。例如，您可以创建针对于特定应用程序的对象，如实例、存储桶和负载均衡器，但不能创建与网络相关的组件，如 VPC 、子网或入站规则。

您在创建集群时使用的 AWS 凭证不需要 VPC 和 VPC 中的核心网络组件（如子网、路由表、互联网网关、NAT 和 VPN）所需的网络权限。您仍然需要获取集群中的机器需要的应用程序资源的权限，如 ELB 、安全组、S3 存储桶和节点。

2.8.3.4. 集群间隔离

如果您将 OpenShift Container Platform 部署到现有网络中，集群服务的隔离将在以下方面减少：

- 您可以在同一 VPC 中安装多个 OpenShift Container Platform 集群。
- 整个网络允许 ICMP 入站流量。
- 整个网络都允许 TCP 22 入站流量 (SSH)。
- 整个网络都允许 control plane TCP 6443 入站流量 (Kubernetes API)。
- 整个网络都允许 control plane TCP 22623 入站流量 (MCS) 。

2.8.4. OpenShift Container Platform 的互联网访问

在 OpenShift Container Platform 4.6 中，您需要访问互联网来安装集群。

您必须具有以下互联网访问权限：

- 访问 OpenShift Cluster Manager 以下载安装程序并执行订阅管理。如果集群可以访问互联网，并且没有禁用 Telemetry，该服务会自动授权您的集群。
- 访问 Quay.io，以获取安装集群所需的软件包。
- 获取执行集群更新所需的软件包。

重要

如果您的集群无法直接访问互联网，则可以在置备的某些类基础架构上执行受限网络安装。在此过程中，您要下载所需的内容，并使用它在镜像 registry（mirror registry） 中填充安装集群并生成安装程序所需的软件包。对于某些安装类型，集群要安装到的环境不需要访问互联网。在更新集群之前，要更新 registry 镜像系统中的内容。

2.8.5. 生成 SSH 私钥并将其添加到代理中

如果要在集群上执行安装调试或灾难恢复，则必须为 **ssh-agent** 和安装程序提供 SSH 密钥。您可以使用此密钥访问公共集群中的 bootstrap 机器来排除安装问题。

注意

在生产环境中，您需要进行灾难恢复和调试。

您可以使用此密钥以 **core** 用户身份通过 SSH 连接到 master 节点。在部署集群时，此密钥会添加到 **core** 用户的 **~/.ssh/authorized_keys** 列表中。

注意

您必须使用一个本地密钥，而不要使用在特定平台上配置的密钥，如 AWS 密钥对。

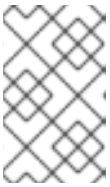
流程

1. 如果还没有为计算机上免密码身份验证而配置的 SSH 密钥，请创建一个。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ ssh-keygen -t ed25519 -N '' \
  -f <path>/<file_name> 1
```

1 指定新 SSH 密钥的路径和文件名，如 **~/.ssh/id_rsa**。如果您已有密钥对，请确保您的公钥位于 **~/.ssh** 目录中。

运行此命令会在指定的位置生成不需要密码的 SSH 密钥。



注意

如果您计划在 **x86_64** 架构中安装使用 FIPS 验证的/Modules in Process 加密库的 OpenShift Container Platform 集群，不要创建使用 **ed25519** 算法的密钥。反之，创建一个使用 **rsa** 或 **ecdsa** 算法的密钥。

2. 作为后台任务启动 **ssh-agent** 进程：

```
$ eval "$(ssh-agent -s)"
```

输出示例

```
Agent pid 31874
```

注意

如果您的集群采用 FIPS 模式，则只使用 FIPS 兼容算法来生成 SSH 密钥。密钥必须是 RSA 或 ECDSA。

3. 将 SSH 私钥添加到 **ssh-agent**：

```
$ ssh-add <path>/<file_name> 1
```

输出示例

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

1 指定 SSH 私钥的路径和文件名，如 **~/.ssh/id_rsa**

后续步骤

- 在安装 OpenShift Container Platform 时，为安装程序提供 SSH 公钥。

2.8.6. 获取安装程序

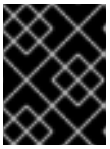
在安装 OpenShift Container Platform 之前，将安装文件下载到本地计算机上。

先决条件

- 运行 Linux 或 macOS 的计算机，本地磁盘空间为 500 MB

流程

1. 访问 OpenShift Cluster Manager 站点的 Infrastructure Provider 页面。如果您有红帽帐号，请使用自己的凭证登录。如果没有，请创建一个帐户。
2. 选择您的基础架构供应商。
3. 进入适用于您的安装类型的页面，下载您的操作系统的安装程序，并将文件放在要保存安装配置文件的目录中。。



重要

安装程序会在用来安装集群的计算机上创建若干文件。在完成集群安装后，您必须保留安装程序和安装程序所创建的文件。这两个文件都需要删除集群。

重要

删除安装程序创建的文件不会删除您的集群，即使集群在安装过程中失败也是如此。要删除集群，为特定云供应商完成 OpenShift Container Platform 卸载流程。

4. 提取安装程序。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ tar xvf openshift-install-linux.tar.gz
```

5. 从 Red Hat OpenShift Cluster Manager 下载安装 pull secret 。通过此 pull secret，您可以进行所含授权机构提供的服务的身份验证，这些服务包括为 OpenShift Container Platform 组件提供容器镜像的 Quay.io。

2.8.7. 手动创建安装配置文件

对于只能从内部网络访问且不能在互联网中看到的私有 OpenShift Container Platform 集群安装，您必须手动生成安装配置文件。

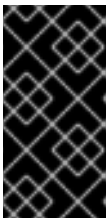
先决条件

- 获取 OpenShift Container Platform 安装程序和集群的访问令牌。

流程

1. 创建用来存储您所需的安装资产的安装目录：

```
$ mkdir <installation_directory>
```



重要

您必须创建目录。一些安装信息，如 bootstrap X.509 证书，有较短的过期间隔，因此不要重复使用安装目录。如果要重复使用另一个集群安装中的个别文件，可以将其复制到您的目录中。但是，一些安装数据的文件名可能会在发行版本之间有所改变。从 OpenShift Container Platform 老版本中复制安装文件时要格外小心。

2. 自定义以下 **install-config.yaml** 文件模板，并将它保存到 **<installation_directory>** 中。

注意

此配置文件必须命名为 **install-config.yaml**。

3. 备份 **install-config.yaml** 文件，以便用于安装多个集群。

重要

install-config.yaml 文件会在安装过程的下一步骤中消耗掉。现在必须备份它。

2.8.7.1. 安装配置参数

在部署 OpenShift Container Platform 集群前，您可以提供参数值，以描述托管集群的云平台的帐户并选择性地自定义集群平台。在创建 **install-config.yaml** 安装配置文件时，您可以通过命令行来提供所需的参数的值。如果要自定义集群，可以修改 **install-config.yaml** 文件来提供关于平台的更多信息。

注意

安装之后，您无法修改 **install-config.yaml** 文件中的这些参数。

重要

openshift-install 命令不验证参数的字段名称。如果指定了不正确的名称，则不会创建相关的文件或对象，且不会报告错误。确保所有指定的参数的字段名称都正确。

2.8.7.1.1. 所需的配置参数

下表描述了所需的安装配置参数：

表 2.22. 所需的参数

参数	描述	值
apiVersion	**install-config.yaml** 内容的 API 版本。当前版本是 **v1**。安装程序还可能支持旧的 API 版本。	字符串

参数	描述	值
baseDomain	云供应商的基域。此基础域用于创建到 OpenShift Container Platform 集群组件的路由。集群的完整 DNS 名称是 **baseDomain** 和 **metadata.name** 参数值的组合，其格式为 **<metadata.name>.<baseDomain>**。	完全限定域名或子域名，如 **example.com**。
metadata	Kubernetes 资源 **ObjectMeta**，其中只消耗 **name** 参数。	对象
metadata.name	集群的名称。集群的 DNS 记录是 **{{.metadata.name}}.{{.baseDomain}}** 的子域。	小写字母,连字符(-)和句点(.)的字符串，如 **dev**。
platform	执行安装的具体平台配置：**aws**、**baremetal**、**azure**、**openstack**、**ovirt**、**vsphere**。有关 **platform.<platform>** 参数的额外信息，请参考下表来了解您的具体平台。	对象

pullSecret	从 Red Hat OpenShift Cluster Manager 获取 pull secret，验证从 Quay.io 等服务中下载 OpenShift Container Platform 组件的容器镜像。	{
		"auths":{
		"cloud.openshift.com":{
		"auth":"b3Blb=",
		"email":"you@example.com"
		},
		"quay.io":{
		"auth":"b3Blb=",
		"email":"you@example.com"
		}
		}
		}

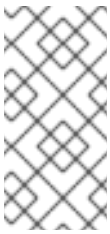
2.8.7.1.2. 网络配置参数

您可以根据现有网络基础架构的要求自定义安装配置。例如，您可以扩展集群网络的 IP 地址块，或者提供不同于默认值的不同 IP 地址块。

只支持 IPv4 地址。

表 2.23. 网络参数

参数	描述	值
networking	集群网络的配置。	对象  **注意** 您不能在安装后修改 **networking** 对象指定的参数。
networking.networkType	要安装的集群网络供应商 Container Network Interface（CNI）插件。	**OpenShiftSDN** 或 **OVNKubernetes**。默认值为 **OpenShiftSDN**。
networking.clusterNetwork	pod 的 IP 地址块。 默认值为 **10.128.0.0/14**，主机前缀为 **/23**。 如果您指定多个 IP 地址块，则块不得互相重叠。	一个对象数组。例如： networking: clusterNetwork: - cidr: 10.128.0.0/14 hostPrefix: 23
networking.clusterNetwork.cidr	使用 **networking.clusterNetwork** 时需要此项。IP 地址块。 一个 IPv4 网络。	使用 CIDR 形式的 IP 地址块。IPv4 块的前缀长度介于 **0** 到 **32** 之间。
networking.clusterNetwork.hostPrefix	分配给每个单独节点的子网前缀长度。例如，如果 **hostPrefix** 设为 **23**，则每个节点从所给的 **cidr** 中分配一个 **/23** 子网。**hostPrefix** 值 **23** 提供 510（2^(32 - 23)- 2）个 pod IP 地址。	子网前缀。 默认值为 **23**。
networking.serviceNetwork	服务的 IP 地址块。默认值为 **172.30.0.0/16**。 OpenShift SDN 和 OVN-Kubernetes 网络供应商只支持服务网络的一个 IP 地址块。	CIDR 格式具有 IP 地址块的数组。例如： networking: serviceNetwork: - 172.30.0.0/16
networking.machineNetwork	机器的 IP 地址块。 如果您指定多个 IP 地址块，则块不得互相重叠。	一个对象数组。例如： networking: machineNetwork: - cidr: 10.0.0.0/16

参数	描述	值
networking.machineNetwork.cidr	使用 **networking.machineNetwork** 时需要。IP 地址块。libvirt 以外的所有平台的默认值为 **10.0.0.0/16**。对于 libvirt，默认值为 **192.168.126.0/24**。	CIDR 表示法中的 IP 网络块。 例如： **10.0.0.0/16**。  **注意** 将 **networking.machineNetwork** 设置为与首选 NIC 所在的 CIDR 匹配。

2.8.7.1.3. 可选配置参数

下表描述了可选安装配置参数：

表 2.24. 可选参数

参数	描述	值
additionalTrustBundle	添加到节点可信证书存储中的 PEM 编码 X.509 证书捆绑包。配置了代理时，也可以使用这个信任捆绑包。	字符串
compute	组成计算节点的机器的配置。	machine-pool 对象的数组。详情请查看以下"Machine-pool"表。
compute.architecture	决定池中机器的指令集合架构。目前不支持异构集群，因此所有池都必须指定相同的架构。有效值为 **amd64** （默认值）。	字符串
compute.hyperthreading	是否在计算机器上启用或禁用并发多线程或**超线程**。默认情况下，启用并发多线程以提高机器内核的性能。  **重要** 如果禁用并发多线程，请确保在容量规划时考虑到机器性能可能会显著降低的问题。	**Enabled** 或 **Disabled**
compute.name	使用 **compute** 时需要此值。机器池的名称。	**worker**

参数	描述	值
compute.platform	使用 **compute** 时需要此值。使用此参数指定托管 worker 机器的云供应商。此参数值必须与 **controlPlane.platform** 参数值匹配。	**aws**、**azure**、**gcp**、**openstack**、**ovirt**、**vsphere** 或 **{}**
compute.replicas	要置备的计算机器数量，也称为 worker 机器。	大于或等于 **2** 的正整数。默认值为 **3**。
controlPlane	组成 control plane 的机器的配置。	**MachinePool** 对象的数组。详情请查看以下"Machine-pool"表。
controlPlane.architecture	决定池中机器的指令集合架构。目前不支持异构集群，因此所有池都必须指定相同的架构。有效值为 **amd64** （默认值）。	字符串
controlPlane.hyperthreading	是否在 control plane 机器上启用或禁用并发多线程或**超线程**。默认情况下，启用并发多线程以提高机器内核的性能。  **重要** 如果禁用并发多线程，请确保在容量规划时考虑到机器性能可能会显著降低的问题。	**Enabled** 或 **Disabled**
controlPlane.name	使用 **controlPlane** 时需要。机器池的名称。	**master**
controlPlane.platform	使用 **controlPlane** 时需要。使用此参数指定托管 control plane 机器的云供应商。此参数值必须与 **compute.platform** 参数值匹配。	**aws**、**azure**、**gcp**、**openstack**、**ovirt**、**vsphere** 或 **{}**
controlPlane.replicas	要置备的 control plane 机器数量。	唯一支持的值是 **3**，它是默认值。

参数	描述	值
credentialsMode	Cloud Credential Operator（CCO）模式。如果没有指定任何模式，CCO 会动态地尝试决定提供的凭证的功能，在支持多个模式的平台上使用 mint 模式。  **注意** 不是所有 CCO 模式都支持所有云供应商。如需有关 CCO 模式的更多信息，请参阅*Red Hat Operator 参考指南*内容中的 *Cloud Credential Operator* 条目。	**Mint**、**Passthrough**、**Manual** 或空字符串(**""**)。
fips	启用或禁用 FIPS 模式。默认为 **false**（禁用）。如果启用了 FIPS 模式，运行 OpenShift Container Platform 的 Red Hat Enterprise Linux CoreOS (RHCOS) 机器会绕过默认的 Kubernetes 加密套件，并使用由 RHCOS 提供的加密模块。  **重要** 只有在 **x86_64** 架构中的 OpenShift Container Platform 部署支持 FIPS 验证的/Modules in Process 加密库。  **注意** 如果使用 Azure File 存储，则无法启用 FIPS 模式。	**false** 或 **true**
imageContentSources	release-image 内容的源和仓库。	对象数组。包括一个 **source** 以及可选的 **mirrors**，如下表所示。
imageContentSources.source	使用 **imageContentSources** 时需要。指定用户在镜像拉取规格中引用的仓库。	字符串
imageContentSources.mirrors	指定可能还包含同一镜像的一个或多个仓库。	字符串数组

参数	描述	值
publish	如何发布或公开集群的面向用户的端点，如 Kubernetes API、OpenShift 路由。	**Internal** 或 **External**。把 **publish** 设置为 **Internal** 以部署一个私有集群，它不能被互联网访问。默认值为 **External**。
sshKey	用于验证集群机器访问的 SSH 密钥或密钥。  **注意** 对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 **ssh-agent** 进程使用的 SSH 密钥。	一个或多个密钥。例如： sshKey: <key1> <key2> <key3>

2.8.7.1.4. 可选的 AWS 配置参数

下表描述了可选的 AWS 配置参数：

表 2.25. 可选的 AWS 参数

参数	描述	值
compute.platform.aws.amiID	用于为集群引导计算机器的 AWS AMI。对于需要自定义 RHCOS AMI 的区域来说，这是必需的。	属于集合 AWS 区域的任何已发布或自定义 RHCOS AMI。
compute.platform.aws.rootVolume.iops	为根卷保留的每秒输入/输出操作 (IOPS) 数。	整数，如 **4000**。
compute.platform.aws.rootVolume.size	以 GiB 为单位的根卷大小。	整数，如 **500**。
compute.platform.aws.rootVolume.type	根卷的类型。	有效的 AWS EBS 卷类型，如 **io1**。
compute.platform.aws.rootVolume.kmsKeyARN	KMS 密钥的 Amazon 资源名称（密钥 ARN）。这是使用特定 KMS 密钥加密 worker 节点的操作系统卷。	有效的 密钥 ID 或密钥 ARN。

参数	描述	值
compute.platform.aws.type	计算机器的 EC2 实例类型。	有效的 AWS 实例类型，如 **c5.9xlarge**。
compute.platform.aws.zones	安装程序在其中为计算机机器池创建机器的可用区。如果您提供自己的 VPC，则必须在那个可用域中提供一个子网。	有效 AWS 可用区的列表，如 **us-east-1c**，以 YAML 序列表示。
compute.aws.region	安装程序在其中创建计算资源的 AWS 区域。	任何有效的 AWS 区域，如 **us-east-1**。
controlPlane.platform.aws.amiID	用于为集群引导 control plane 机器的 AWS AMI。对于需要自定义 RHCOS AMI 的区域来说，这是必需的。	属于集合 AWS 区域的任何已发布或自定义 RHCOS AMI。
controlPlane.platform.aws.rootVolume.kmsKeyARN	KMS 密钥的 Amazon 资源名称（密钥 ARN）。这需要使用特定的 KMS 密钥加密 control plane 节点的操作系统卷。	有效的 密钥 ID 和密钥 ARN。
controlPlane.platform.aws.type	control plane 机器的 EC2 实例类型。	有效的 AWS 实例类型，如 **c5.9xlarge**。
controlPlane.platform.aws.zones	安装程序在其中为 control plane 机器池创建机器的可用区。	有效 AWS 可用区的列表，如 **us-east-1c**，以 YAML 序列表示。
controlPlane.aws.region	安装程序在其中创建 control plane 资源的 AWS 区域。	有效的 AWS 区域，如 **us-east-1**。
platform.aws.amiID	用于为集群引导所有机器的 AWS AMI。如果设置，AMI 必须属于与集群相同的区域。对于需要自定义 RHCOS AMI 的区域来说，这是必需的。	属于集合 AWS 区域的任何已发布或自定义 RHCOS AMI。
platform.aws.serviceEndpoints.name	AWS 服务端点名称。只有在必须使用替代 AWS 端点（如 FIPS）时，才需要自定义端点。可以为 EC2、S3、IAM、Elastic Load Balancing、Tagging、Route 53 和 STS AWS 服务指定自定义 API 端点。	有效的 AWS 服务端点名称。
platform.aws.serviceEndpoints.url	AWS 服务端点 URL。URL 必须使用 **https** 协议，主机必须信任该证书。	有效的 AWS 服务端点 URL。

参数	描述	值
platform.aws.userTags	键与值的映射，安装程序将其作为标签添加到它所创建的所有资源。	任何有效的 YAML 映射，如 **<key>: <value>** 格式的键值对。如需有关 AWS 标签的更多信息，请参阅 AWS 文档中的标记您的 Amazon EC2 资源。
platform.aws.subnets	如果您提供 VPC，而不是让安装程序为您创建 VPC，请指定要使用的集群子网。子网必须是您指定的同一 **machineNetwork[].cidr** 范围的一部分。对于标准集群，为每个可用区指定一个公共和私有子网。对于私有集群，为每个可用区指定一个私有子网。	有效的子网 ID。

2.8.7.2. AWS 的自定义 install-config.yaml 文件示例

您可以自定义 **install-config.yaml** 文件，以指定有关 OpenShift Container Platform 集群平台的更多信息，或修改所需参数的值。

重要

此示例 YAML 文件仅供参考。您必须使用安装程序来获取 **install-config.yaml** 文件，并且修改该文件。

```
apiVersion: v1
baseDomain: example.com (1)
credentialsMode: Mint (2)
controlPlane: (3) (4)
  hyperthreading: Enabled (5)
  name: master
  platform:
    aws:
      zones:
      - us-west-2a
      - us-west-2b
      rootVolume:
        iops: 4000
        size: 500
        type: io1 (6)
      type: m5.xlarge
  replicas: 3
compute: (7)
- hyperthreading: Enabled (8)
  name: worker
  platform:
    aws:
      rootVolume:
        iops: 2000
```

```
        size: 500
        type: io1 9
      type: c5.4xlarge
      zones:
      - us-west-2c
  replicas: 3
metadata:
  name: test-cluster 10
networking:
  clusterNetwork:
  - cidr: 10.128.0.0/14
    hostPrefix: 23
  machineNetwork:
  - cidr: 10.0.0.0/16
  networkType: OpenShiftSDN
  serviceNetwork:
  - 172.30.0.0/16
platform:
  aws:
    region: us-west-2 11
    userTags:
      adminContact: jdoe
      costCenter: 7536
    subnets: 12
    - subnet-1
    - subnet-2
    - subnet-3
    amiID: ami-96c6f8f7 13
    serviceEndpoints: 14
      - name: ec2
        url: https://vpce-id.ec2.us-west-2.vpce.amazonaws.com
    hostedZone: Z3URY6TWQ91KVV 15
fips: false 16
sshKey: ssh-ed25519 AAAA... 17
publish: Internal 18
pullSecret: '{"auths": ...}' 19
```

1 10 11 19 必需。安装程序会提示您输入这个值。

2 可选：添加此参数来强制 Cloud Credential Operator（CCO）使用指定的模式，而不是让 CCO 动态尝试决定凭证的功能。如需有关 CCO 模式的详情，请参阅 *Red Hat Operator* 参考内容中的 *Cloud Credential Operator* 条目。

3 7 如果没有提供这些参数和值，安装程序会提供默认值。

4 **controlPlane** 部分是一个单映射，但 **compute** 部分是一系列映射。为满足不同数据结构的要求，**compute** 部分的第一行必须以连字符 **-** 开头，**controlPlane** 部分的第一行则不可以连字符开头。只使用一个 control plane 池。

5 8 是否要启用或禁用并发多线程或**超线程**。默认情况下，启用并发多线程以提高机器内核的性能。您可以通过将参数值设为 **Disabled** 来禁用。如果您在某些集群机器上禁用并发多线程，则必须在所有集群机器上禁用。

重要

如果禁用并发多线程，请确保在容量规划时考虑到机器性能可能会显著降低的问题。如果您对机器禁用并发多线程，请使用较大的实例类型，如 **m4.2xlarge** 或 **m5.2xlarge**。

6 9 要为 etcd 配置更快的存储，特别是对于较大的集群，请将存储类型设置为 **io1**，并将 **iops** 设为 **2000**。

12 如果您提供自己的 VPC，为集群使用的每个可用区指定子网。

13 用于为集群引导机器的 AMI ID。如果设置，AMI 必须属于与集群相同的区域。

14 AWS 服务端点。在安装到未知 AWS 区域时，需要自定义端点。端点 URL 必须使用 **https** 协议，主机必须信任该证书。

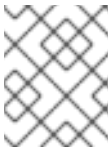
15 您现有 Route 53 私有托管区的 ID。提供现有的托管区需要您提供自己的 VPC，托管区已在安装集群前与 VPC 关联。如果未定义，安装程序会创建一个新的托管区。

16 是否启用或禁用 FIPS 模式。默认情况下不启用 FIPS 模式。如果启用了 FIPS 模式，运行 OpenShift Container Platform 的 Red Hat Enterprise Linux CoreOS (RHCOS) 机器会绕过默认的 Kubernetes 加密套件，并使用由 RHCOS 提供的加密模块。

重要

只有在 **x86_64** 架构中的 OpenShift Container Platform 部署支持 FIPS 验证的/Modules in Process 加密库。

17 您可以选择提供您用来访问集群中机器的 **sshKey** 值。



注意

对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 **ssh-agent** 进程使用的 SSH 密钥。

18 如何发布集群的面向用户的端点。把 **publish** 设置为 **Internal** 以部署一个私有集群，它不能被互联网访问。默认值为 **External**。

2.8.7.3. 在安装过程中配置集群范围代理

生产环境可能会拒绝直接访问互联网，而是提供 HTTP 或 HTTPS 代理。您可以通过在 **install-config.yaml** 文件中配置代理设置，将新的 OpenShift Container Platform 集群配置为使用代理。

先决条件

- 您有一个现有的 **install-config.yaml** 文件。
- 您检查了集群需要访问的站点，并决定是否需要绕过代理。默认情况下代理所有集群出口流量，包括对托管云供应商 API 的调用。您需要将站点添加到 **Proxy** 对象的 **spec.noProxy** 字段来绕过代理。

注意

Proxy 对象 **status.noProxy** 字段使用安装配置中的 **networking.machineNetwork[].cidr**、**networking.clusterNetwork[].cidr** 和 **networking.serviceNetwork[]** 字段的值填充。

对于在 Amazon Web Services(AWS)、Google Cloud Platform(GCP)、Microsoft Azure 和 Red Hat OpenStack Platform(RHOSP)上安装, **Proxy** 对象 **status.noProxy** 字段也会使用实例元数据端点填充(**169.254.169.254**)。

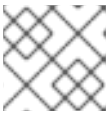
- 如果您的集群位于 AWS 上，请将 **ec2.<region>.amazonaws.com**、**elasticloadbalancing.<region>.amazonaws.com** 和 **s3.<region>.amazonaws.com** 端点添加到 VPC 端点。需要这些端点才能完成节点到 AWS EC2 API 的请求。由于代理在容器级别而不是节点级别工作，因此您必须通过 AWS 专用网络将这些请求路由到 AWS EC2 API。在代理服务器中的允许列表中添加 EC2 API 的公共 IP 地址是不够的。

流程

1. 编辑 **install-config.yaml** 文件并添加代理设置。例如：

```
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: example.com 3
additionalTrustBundle: | 4
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
...
```

1. 用于创建集群外 HTTP 连接的代理 URL。URL 必须是 **http**。
2. 用于创建集群外 HTTPS 连接的代理 URL。
3. 要排除在代理中的目标域名、IP 地址或其他网络 CIDR 的逗号分隔列表。在域前面加 **.** 来仅匹配子域。例如： **.y.com** 匹配 **x.y.com**，但不匹配 **y.com**。使用 ***** 绕过所有目的地的代理。
4. 如果提供，安装程序会在 **openshift-config 命名空间中生成名为 user-ca- bundle** 的配置映射来保存额外的 CA 证书。如果您提供 **additionalTrustBundle** 和至少一个代理设置，则 **Proxy** 对象会被配置为引用 **trustedCA** 字段中的 **user-ca-bundle** 配置映射。然后，Cluster Network Operator 会创建一个 **trusted-ca-bundle** 配置映射，该配置映射将为 **trustedCA** 参数指定的内容与 RHCOS 信任捆绑包合并。 **additionalTrustBundle** 字段是必需的，除非代理的身份证书由来自 RHCOS 信任捆绑包的颁发机构签名。

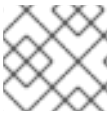


注意

安装程序不支持代理的 **readinessEndpoints** 字段。

2. 保存该文件，并在安装 OpenShift Container Platform 时引用。

安装程序会创建一个名为 **cluster** 的集群范围代理，该代理使用提供的 **install-config.yaml** 文件中的代理设置。如果没有提供代理设置，仍然会创建一个 **cluster Proxy** 对象，但它会有一个空 **spec**。



注意

只支持名为 **cluster** 的 **Proxy** 对象，且无法创建额外的代理。

2.8.8. 部署集群

您可以在兼容云平台中安装 OpenShift Container Platform。

重要

安装程序的 **create cluster** 命令只能在初始安装过程中运行一次。

先决条件

- 配置托管集群的云平台的帐户。
- 获取 OpenShift Container Platform 安装程序以及集群的 pull secret。

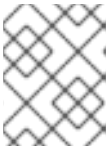
流程

1. 更改为包含安装程序的目录并初始化集群部署：

```
$ ./openshift-install create cluster --dir <installation_directory> \ 1
    --log-level=info 2
```

1 对于 **<installation_directory>**，请指定

2 要查看不同的安装详情，请指定 **warn**、**debug** 或 **error**，而不要指定 **info**。



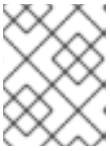
注意

如果您在主机上配置的云供应商帐户没有足够的权限来部署集群，安装过程将会停止，并且显示缺少的权限。

集群部署完成后，终端会显示访问集群的信息，包括指向其 Web 控制台的链接和 **kubeadmin** 用户的凭证。

输出示例

```
...
INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run 'export
KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'
INFO Access the OpenShift web-console here: https://console-openshift-
console.apps.mycluster.example.com
INFO Login to the console with user: "kubeadmin", and password: "4vYBz-Ee6gm-ymBZj-
Wt5AL"
INFO Time elapsed: 36m22s
```



注意

当安装成功时，集群访问和凭证信息还会输出到 **<installation_directory>/.openshift_install.log**。

重要

- 安装程序生成的 Ignition 配置文件包含在 24 小时后过期的证书，然后在过期时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外情况是，您需要手动批准待处理的 **node-bootstrapper** 证书签名请求（CSR）来恢复 kubelet 证书。如需更多信息，请参阅*从过期的 control plane 证书中恢复*的文档。
- 建议您在生成 12 小时后使用 Ignition 配置文件，因为集群安装后 24 小时证书从 16 小时轮转至 22 小时。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中运行证书更新时避免安装失败。

重要

您不得删除安装程序或安装程序所创建的文件。需要这两者才能删除集群。

2.8.9. 通过下载二进制文件安装 OpenShift CLI

您需要安装 CLI（**oc**） 来使用命令行界面与 OpenShift Container Platform 进行交互。您可在 Linux、Windows 或 macOS 上安装 **oc**。

重要

如果安装了旧版本的 **oc**，则无法使用 OpenShift Container Platform 4.6 中的所有命令。下载并安装新版本的 **oc**。

2.8.9.1. 在 Linux 上安装 OpenShift CLI

您可以按照以下流程在 Linux 上安装 OpenShift CLI（**oc**）二进制文件。

流程

1. 进入到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 在 **Version** 下拉菜单中选择相应的版本。
3. 单击 **OpenShift v4.6 Linux 客户端** 条目旁边的 **Download Now**，再保存文件。
4. 解包存档：

```
$ tar xvzf <file>
```

5. 把 **oc** 二进制代码放到 **PATH** 中的目录中。
   执行以下命令可以查看当前的 **PATH** 设置：

```
$ echo $PATH
```

安装 OpenShift CLI 后，可以使用 **oc** 命令：

-

```
$ oc <command>
```

2.8.9.2. 在 Windows 上安装 OpenShift CLI

您可以按照以下流程在 Windows 上安装 OpenShift CLI（**oc**）二进制代码。

流程

1. 进入到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 在 **Version** 下拉菜单中选择相应的版本。
3. 单击 **OpenShift v4.6 Windows 客户端**条目旁边的 **Download Now**，再保存文件。
4. 使用 ZIP 程序解压存档。
5. 把 **oc** 二进制代码放到 **PATH** 中的目录中。
   要查看您的 **PATH**，请打开命令提示窗口并执行以下命令：

   ```
   C:\> path
   ```

安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
C:\> oc <command>
```

2.8.9.3. 在 macOS 上安装 OpenShift CLI

您可以按照以下流程在 macOS 上安装 OpenShift CLI（**oc**）二进制代码。

流程

1. 进入到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 在 **Version** 下拉菜单中选择相应的版本。
3. 单击 **OpenShift v4.6 MacOSX 客户端**条目旁边的 **Download Now**，再保存文件。
4. 解包和解压存档。
5. 将 **oc** 二进制文件移到 PATH 的目录中。
   要查看您的 **PATH**，打开一个终端窗口并执行以下命令：

   ```
   $ echo $PATH
   ```

安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
$ oc <command>
```

2.8.10. 使用 CLI 登录到集群

您可以通过导出集群 **kubeconfig** 文件，以默认系统用户身份登录集群。**kubeconfig** 文件包含关于集群的信息，供 CLI 用于将客户端连接到正确集群和 API 服务器。该文件特只适用于一个特定的集群，在 OpenShift Container Platform 安装过程中创建。

先决条件

- 已部署了 OpenShift Container Platform 集群。
- 已安装 **oc** CLI。

流程

1. 导出 **kubeadmin** 凭证：

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

2. 使用导出的配置，验证能否成功运行 **oc** 命令：

```
$ oc whoami
```

输出示例

```
system:admin
```

2.8.11. 使用 Web 控制台登录到集群

kubeadmin 用户默认在 OpenShift Container Platform 安装后存在。您可以使用 OpenShift Container Platform Web 控制台以 **kubeadmin** 用户身份登录集群。

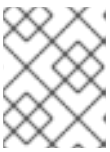
先决条件

- 有访问安装主机的访问权限。
- 您完成了集群安装，所有集群 Operator 都可用。

流程

1. 从安装主机上的 **kubeadmin -password** 文件中获取 kubeadmin 用户的密码：

```
$ cat <installation_directory>/auth/kubeadmin-password
```

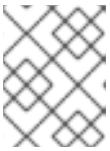


注意

另外，您还可以从安装主机上的 **<installation_directory>/.openshift_install.log** 日志文件获取 **kubeadmin** 密码。

2. 列出 OpenShift Container Platform Web 控制台路由：

```
$ oc get routes -n openshift-console | grep 'console-openshift'
```



注意

另外，您还可以从安装主机上的 **<installation_directory>/.openshift_install.log 日志** 文件获取 OpenShift Container Platform 路由。

输出示例

```
console     console-openshift-console.apps.<cluster_name>.<base_domain>          console
https   reencrypt/Redirect   None
```

3. 在 Web 浏览器中导航到上一命令输出中包括的路由，以 **kubeadmin** 用户身份登录。

其他资源

- 如需有关访问和了解 OpenShift Container Platform Web 控制台的更多信息，请参阅访问 Web 控制台。

2.8.12. OpenShift Container Platform 的 Telemetry 访问

在 OpenShift Container Platform 4.6 中，默认运行的 Telemetry 服务提供有关集群健康状况和成功更新的指标，需要访问互联网。如果您的集群连接到互联网，Telemetry 会自动运行，而且集群会注册到 OpenShift Cluster Manager。

确认 OpenShift Cluster Manager 清单正确后，可以由 Telemetry 自动维护，也可以使用 OpenShift Cluster Manager 手动维护，使用订阅监控来跟踪帐户或多集群级别的 OpenShift Container Platform 订阅。

其他资源

- 有关 Telemetry 服务的更多信息，请参阅关于远程健康监控。

2.8.13. 后续步骤

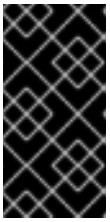
- 验证安装。
- 自定义集群。
- 如果需要，您可以选择不使用远程健康报告。
- 如果需要，您可以删除云供应商凭证。

2.9. 在 AWS 上将集群安装到一个政府区域

在 OpenShift Container Platform 版本 4.6 中，您可以在 Amazon Web Services（AWS）上将集群安装到一个政府区域。要配置政府区域，请在安装集群前修改 **install-config.yaml** 文件中的参数。

2.9.1. 先决条件

- 查看有关 OpenShift Container Platform 安装和更新流程的详细信息。
- 配置 AWS 帐户以托管集群。



重要

如果您的计算机上存储有 AWS 配置集，则不要在使用多因素验证设备的同时使用您生成的临时会话令牌。在集群的整个生命周期中，集群会持续使用您的当前 AWS 凭证来创建 AWS 资源，因此您必须使用长期凭证。要生成适当的密钥，请参阅 AWS 文档中的管理 IAM 用户的访问密钥。您可在运行安装程序时提供密钥。

- 如果使用防火墙，则必须将其配置为允许集群需要访问的站点。
- 如果不允许系统管理身份和访问管理（IAM），集群管理员可以 手动创建和维护 IAM 凭证。手动模式也可以用于云 IAM API 无法访问的环境中。

2.9.2. AWS 政府区域

OpenShift Container Platform 支持将集群部署到 AWS GovCloud(US) 区域。AWS GovCloud 是为需要运行敏感负载的美国政府机构、企业、企业和其他美国客户特别设计的。

这些区域尚未发布 Red Hat Enterprise Linux CoreOS（RHCOS）Amazon Machine Images（AMI），因此您必须上传属于该区的自定义 AMI。

支持以下 AWS GovCloud 分区：

- **us-gov-west-1**
- **us-gov-east-1**

由于那些区域的 RHCOS AMI 不是由红帽提供的，所以必须在 **install-config.yaml** 文件中手动配置 AWS GovCloud 区域和自定义 AMI。

2.9.3. 私有集群

您可以部署不公开外部端点的私有 OpenShift Container Platform 集群。私有集群只能从内部网络访问，且无法在互联网中看到。

注意

AWS GovCloud 的 Route 53 不支持公共区。因此，如果集群部署到 AWS 政府区域，集群必须是私有的。

默认情况下，OpenShift Container Platform 被置备为使用可公开访问的 DNS 和端点。私有集群在部署集群时将 DNS 、Ingress Controller 和 API 服务器设置为私有。这意味着，集群资源只能从您的内部网络访问，且不能在互联网中看到。

要部署私有集群，您必须使用符合您的要求的现有网络。您的集群资源可能会在网络中的其他集群间共享。

另外，您必须从可访问您置备的云的 API 服务、您置备的网络上的主机以及可以连接到互联网来获取安装介质的机器上部署私有集群。您可以使用符合这些访问要求的机器，并按照您的公司规定进行操作。例如，该机器可以是云网络中的堡垒主机，也可以是可通过 VPN 访问网络的机器。

2.9.3.1. AWS 中的私有集群

要在 Amazon Web Services (AWS) 上创建私有集群，您必须提供一个现有的私有 VPC 和子网来托管集群。安装程序还必须能够解析集群所需的 DNS 记录。安装程序将 Ingress Operator 和 API 服务器配置为只可以从私有网络访问。

集群仍然需要访问互联网来访问 AWS API。

安装私有集群时不需要或创建以下项目：

- 公共子网
- 支持公共入口的公共负载均衡器
- 与集群的 **baseDomain** 匹配的公共 Route 53 区域

安装程序会使用您指定的 **baseDomain** 来创建专用的 Route 53 区域以及集群所需的记录。集群被配置，以便 Operator 不会为集群创建公共记录，且所有集群机器都放置在您指定的私有子网中。

2.9.3.1.1. 限制：

为私有集群添加公共功能的能力有限。

- 在安装后，您无法在不进行额外操作的情况下公开 Kubernetes API 端点。这些额外的操作包括为使用中的每个可用区在 VPC 中创建公共子网，创建公共负载均衡器，以及配置 control plane 安全组以便 6443 端口（Kubernetes API 端口）可以接受来自于互联网的网络流量。
- 如果使用公共服务类型负载均衡器，您必须在每个可用区中为公共子网添加 **kubernetes.io/cluster/<cluster-infra-id>: shared** 标签，以便 AWS 可使用它们来创建公共负载均衡器。

2.9.4. 关于使用自定义 VPC

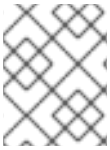
在 OpenShift Container Platform 4.6 中，您可以在 Amazon Web Services（AWS）的现有 Amazon Virtual Private Cloud（VPC）中将集群部署到现有子网中。通过将 OpenShift Container Platform 部署到现有的 AWS VPC 中，您可能会避开新帐户中的限制，或者更容易地利用公司所设置的操作限制。如果您无法获得您自己创建 VPC 所需的基础架构创建权限，请使用这个安装选项。

因为安装程序无法了解您现有子网中还有哪些其他组件，所以无法选择子网 CIDR 。您必须为安装集群的子网配置网络。

2.9.4.1. 使用 VPC 的要求

安装程序不再创建以下组件：

- 互联网网关
- NAT 网关
- 子网
- 路由表
- VPCs
- VPC DHCP 选项
- VPC 端点



注意

安装程序要求您使用由云提供的 DNS 服务器。不支持使用自定义 DNS 服务器，并导致安装失败。

如果您使用自定义 VPC，您必须为安装程序和集群正确配置它及其子网。有关创建和管理 AWS VPC VPC 的更多信息，请参阅 AWS 文档中的 Amazon VPC 控制台向导配置 以及处理 VPC 和子网。

安装程序无法：

- 子类网络范围供群集使用。
- 设置子网的路由表。
- 设置 VPC 选项，如 DHCP。

您必须在安装集群前完成这些任务。有关在 AWS VPC 中配置网络的更多信息，请参阅 VPC 的 VPC 网络组件 和路由表。

您的 VPC 必须满足以下特征：

- VPC 不能使用 **kubernetes.io/cluster/.*: owned** 标签。
  安装程序会修改子网以添加 **kubernetes.io/cluster/.*: shared** 标签，因此您的子网必须至少有一个可用的空闲标签插槽。请参阅 AWS 文档中的 标签限制 部分，以确认安装程序可以为您指定的每个子网添加标签。
- 您必须在 VPC **中启用 enable DnsSupport 和 enableDnsHostnames** 属性，以便集群可以使用附加到 VPC 的 Route 53 区域来解析集群内部 DNS 记录。请参阅 AWS 文档中的您的 VPC 中的 DNS 支持部分 。
  如果您希望使用自己的 Route 53 托管私有区，则必须在安装集群前将现有托管区与 VPC 关联。您可以使用 **install-config.yaml** 文件中的 **platform.aws.hostedZone** 字段定义托管区。
- 如果您使用具有公共访问权限的集群，您必须为每个集群使用的可用区创建一个公共和私有子网。每个可用区不能包含多于一个的公共子网和私有子网。

如果您在断开连接的环境中工作，您将无法访问 EC2 和 ELB 端点的公共 IP 地址。要解决这个问题，您必须创建一个 VPC 端点，并将其附加到集群使用的子网。端点应命名如下：

- **ec2.<region>.amazonaws.com**
- **elasticloadbalancing.<region>.amazonaws.com**
- **s3.<region>.amazonaws.com**

所需的 VPC 组件

您必须提供合适的 VPC 和子网，以便与您的机器通信。

组件	AWS 类型	描述
VPC	• **AWS::EC2::VPC** • **AWS::EC2::VPCEndpoint**	您必须提供一个公共 VPC 供集群使用。VPC 使用引用每个子网的路由表的端点，以改进与托管在 S3 中的 registry 的通信。

<table>
<tr><th>组件</th><th>AWS 类型</th><th colspan="2">描述</th></tr>
<tr><td>公共子网</td><td>AWS::EC2::Subnet
AWS::EC2::SubnetNetworkAclAssociation</td><td colspan="2">您的 VPC 必须有 1 到 3 个可用区的公共子网，并将其与适当的入口规则关联。</td></tr>
<tr><td>互联网网关</td><td>AWS::EC2::InternetGateway
AWS::EC2::VPCGatewayAttachment
AWS::EC2::RouteTable
AWS::EC2::Route
AWS::EC2::SubnetRouteTableAssociation
AWS::EC2::NatGateway
AWS::EC2::EIP</td><td colspan="2">您必须有一个公共互联网网关，以及附加到 VPC 的公共路由。在提供的模板中，每个公共子网都有一个具有 EIP 地址的 NAT 网关。这些 NAT 网关允许集群资源（如专用子网实例）访问互联网，而有些受限网络或代理场景则不需要它们。</td></tr>
<tr><td>网络访问控制</td><td>AWS::EC2::NetworkAcl
AWS::EC2::NetworkAclEntry</td><td colspan="2">您必须允许 VPC 访问下列端口：</td></tr>
<tr><td></td><td></td><td>端口</td><td>原因</td></tr>
<tr><td></td><td></td><td>80</td><td>入站 HTTP 流量</td></tr>
<tr><td></td><td></td><td>443</td><td>入站 HTTPS 流量</td></tr>
<tr><td></td><td></td><td>22</td><td>入站 SSH 流量</td></tr>
<tr><td></td><td></td><td>1024 - 65535</td><td>入站临时流量</td></tr>
<tr><td></td><td></td><td>0 - 65535</td><td>出站临时流量</td></tr>
<tr><td>专用子网</td><td>AWS::EC2::Subnet
AWS::EC2::RouteTable
AWS::EC2::SubnetRouteTableAssociation</td><td colspan="2">您的 VPC 可以具有私有子网。提供的 CloudFormation 模板可为 1 到 3 个可用区创建专用子网。如果您使用专用子网，必须为其提供适当的路由和表。</td></tr>
</table>

2.9.4.2. VPC 验证

要确保您提供的子网适合您的环境，安装程序会确认以下信息：

- 您指定的所有子网都存在。

- 您提供了私有子网。
- 子网 CIDR 属于您指定的机器 CIDR。
- 您为每个可用区提供子网。每个可用区不包含多于一个的公共子网和私有子网。如果您使用私有集群，为每个可用区只提供一个私有子网。否则，为每个可用区提供一个公共和私有子网。
- 您可以为每个私有子网可用区提供一个公共子网。机器不会在没有为其提供私有子网的可用区中置备。

如果您销毁使用现有 VPC 的集群，VPC 不会被删除。从 VPC 中删除 OpenShift Container Platform 集群时，**kubernetes.io/cluster/.*: shared** 标签会从使用它的子网中删除。

2.9.4.3. 权限划分

从 OpenShift Container Platform 4.3 开始，您不需要安装程序置备的基础架构集群部署所需的所有权限。这与您所在机构可能已有的权限划分类似：不同的个人可以在您的云中创建不同的资源。。例如，您可以创建针对于特定应用程序的对象，如实例、存储桶和负载均衡器，但不能创建与网络相关的组件，如 VPC 、子网或入站规则。

您在创建集群时使用的 AWS 凭证不需要 VPC 和 VPC 中的核心网络组件（如子网、路由表、互联网网关、NAT 和 VPN）所需的网络权限。您仍然需要获取集群中的机器需要的应用程序资源的权限，如 ELB 、安全组、S3 存储桶和节点。

2.9.4.4. 集群间隔离

如果您将 OpenShift Container Platform 部署到现有网络中，集群服务的隔离将在以下方面减少：

- 您可以在同一 VPC 中安装多个 OpenShift Container Platform 集群。
- 整个网络允许 ICMP 入站流量。
- 整个网络都允许 TCP 22 入站流量 (SSH)。
- 整个网络都允许 control plane TCP 6443 入站流量 (Kubernetes API)。
- 整个网络都允许 control plane TCP 22623 入站流量 (MCS) 。

2.9.5. OpenShift Container Platform 的互联网访问

在 OpenShift Container Platform 4.6 中，您需要访问互联网来安装集群。

您必须具有以下互联网访问权限：

- 访问 OpenShift Cluster Manager 以下载安装程序并执行订阅管理。如果集群可以访问互联网，并且没有禁用 Telemetry，该服务会自动授权您的集群。
- 访问 Quay.io，以获取安装集群所需的软件包。
- 获取执行集群更新所需的软件包。

重要

如果您的集群无法直接访问互联网，则可以在置备的某些类基础架构上执行受限网络安装。在此过程中，您要下载所需的内容，并使用它在镜像 registry（mirror registry） 中填充安装集群并生成安装程序所需的软件包。对于某些安装类型，集群要安装到的环境不需要访问互联网。在更新集群之前，要更新 registry 镜像系统中的内容。

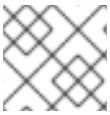
2.9.6. 生成 SSH 私钥并将其添加到代理中

如果要在集群上执行安装调试或灾难恢复，则必须为 **ssh-agent** 和安装程序提供 SSH 密钥。您可以使用此密钥访问公共集群中的 bootstrap 机器来排除安装问题。

注意

在生产环境中，您需要进行灾难恢复和调试。

您可以使用此密钥以 **core** 用户身份通过 SSH 连接到 master 节点。在部署集群时，此密钥会添加到 **core** 用户的 **~/.ssh/authorized_keys** 列表中。



注意

您必须使用一个本地密钥，而不要使用在特定平台上配置的密钥，如 AWS 密钥对。

流程

1. 如果还没有为计算机上免密码身份验证而配置的 SSH 密钥，请创建一个。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ ssh-keygen -t ed25519 -N '' \
  -f <path>/<file_name> 1
```

1 指定新 SSH 密钥的路径和文件名，如 **~/.ssh/id_rsa**。如果您已有密钥对，请确保您的公钥位于 **~/.ssh** 目录中。

运行此命令会在指定的位置生成不需要密码的 SSH 密钥。

注意

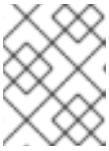
如果您计划在 **x86_64** 架构中安装使用 FIPS 验证的/Modules in Process 加密库的 OpenShift Container Platform 集群，不要创建使用 **ed25519** 算法的密钥。反之，创建一个使用 **rsa** 或 **ecdsa** 算法的密钥。

2. 作为后台任务启动 **ssh-agent** 进程：

```
$ eval "$(ssh-agent -s)"
```

输出示例

```
Agent pid 31874
```



注意

如果您的集群采用 FIPS 模式，则只使用 FIPS 兼容算法来生成 SSH 密钥。密钥必须是 RSA 或 ECDSA。

3. 将 SSH 私钥添加到 **ssh-agent**：

```
$ ssh-add <path>/<file_name> 1
```

输出示例

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

1 指定 SSH 私钥的路径和文件名，如 **~/.ssh/id_rsa**

后续步骤

- 在安装 OpenShift Container Platform 时，为安装程序提供 SSH 公钥。

2.9.7. 获取安装程序

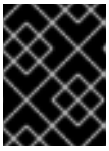
在安装 OpenShift Container Platform 之前，将安装文件下载到本地计算机上。

先决条件

- 运行 Linux 或 macOS 的计算机，本地磁盘空间为 500 MB

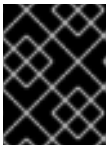
流程

1. 访问 OpenShift Cluster Manager 站点的 Infrastructure Provider 页面。如果您有红帽帐号，请使用自己的凭证登录。如果没有，请创建一个帐户。
2. 选择您的基础架构供应商。
3. 进入适用于您的安装类型的页面，下载您的操作系统的安装程序，并将文件放在要保存安装配置文件的目录中。。



重要

安装程序会在用来安装集群的计算机上创建若干文件。在完成集群安装后，您必须保留安装程序和安装程序所创建的文件。这两个文件都需要删除集群。



重要

删除安装程序创建的文件不会删除您的集群，即使集群在安装过程中失败也是如此。要删除集群，为特定云供应商完成 OpenShift Container Platform 卸载流程。

4. 提取安装程序。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ tar xvf openshift-install-linux.tar.gz
```

5. 从 Red Hat OpenShift Cluster Manager 下载安装 pull secret 。通过此 pull secret，您可以进行所含授权机构提供的服务的身份验证，这些服务包括为 OpenShift Container Platform 组件提供容器镜像的 Quay.io。

2.9.8. 手动创建安装配置文件

在 Amazon Web Services（AWS）上安装 OpenShift Container Platform 时，进入需要自定义 Red Hat Enterprise Linux CoreOS（RHCOS）AMI 的区域时，您必须手动生成安装配置文件。

先决条件

- 获取 OpenShift Container Platform 安装程序和集群的访问令牌。

流程

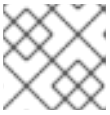
1. 创建用来存储您所需的安装资产的安装目录：

```
$ mkdir <installation_directory>
```

重要

您必须创建目录。一些安装信息，如 bootstrap X.509 证书，有较短的过期间隔，因此不要重复使用安装目录。如果要重复使用另一个集群安装中的个别文件，可以将其复制到您的目录中。但是，一些安装数据的文件名可能会在发行版本之间有所改变。从 OpenShift Container Platform 老版本中复制安装文件时要格外小心。

2. 自定义以下 **install-config.yaml** 文件模板，并将它保存到 **<installation_directory>** 中。



注意

此配置文件必须命名为 **install-config.yaml**。

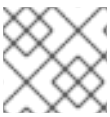
3. 备份 **install-config.yaml** 文件，以便用于安装多个集群。

重要

install-config.yaml 文件会在安装过程的下一步骤中消耗掉。现在必须备份它。

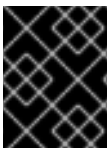
2.9.8.1. 安装配置参数

在部署 OpenShift Container Platform 集群前，您可以提供参数值，以描述托管集群的云平台的帐户并选择性地自定义集群平台。在创建 **install-config.yaml** 安装配置文件时，您可以通过命令行来提供所需的参数的值。如果要自定义集群，可以修改 **install-config.yaml** 文件来提供关于平台的更多信息。



注意

安装之后，您无法修改 **install-config.yaml** 文件中的这些参数。



重要

openshift-install 命令不验证参数的字段名称。如果指定了不正确的名称，则不会创建相关的文件或对象，且不会报告错误。确保所有指定的参数的字段名称都正确。

2.9.8.1.1. 所需的配置参数

下表描述了所需的安装配置参数：

表 2.26. 所需的参数

参数	描述	值
apiVersion	**install-config.yaml** 内容的 API 版本。当前版本是 **v1**。安装程序还可能支持旧的 API 版本。	字符串
baseDomain	云供应商的基域。此基础域用于创建到 OpenShift Container Platform 集群组件的路由。集群的完整 DNS 名称是 **baseDomain** 和 **metadata.name** 参数值的组合，其格式为 **<metadata.name>.<baseDomain>**。	完全限定域名或子域名，如 **example.com**。
metadata	Kubernetes 资源 **ObjectMeta**，其中只消耗 **name** 参数。	对象
metadata.name	集群的名称。集群的 DNS 记录是 **{{.metadata.name}}.{{.baseDomain}}** 的子域。	小写字母,连字符(-)和句点(.)的字符串，如 **dev**。
platform	执行安装的具体平台配置：**aws**、**baremetal**、**azure**、**openstack**、**ovirt**、**vsphere**。有关 **platform.<platform>** 参数的额外信息，请参考下表来了解您的具体平台。	对象

pullSecret	从 Red Hat OpenShift Cluster Manager 获取 pull secret，验证从 Quay.io 等服务中下载 OpenShift Container Platform 组件的容器镜像。	<pre>{
		  "auths":{
		  "cloud.openshift.com":{
		  "auth":"b3Blb=",
		  "email":"you@example.com"
		  },
		  "quay.io":{
		  "auth":"b3Blb=",
		  "email":"you@example.com"
		  }
		  }
		}</pre>

2.9.8.1.2. 网络配置参数

您可以根据现有网络基础架构的要求自定义安装配置。例如，您可以扩展集群网络的 IP 地址块，或者提供不同于默认值的不同 IP 地址块。

只支持 IPv4 地址。

表 2.27. 网络参数

参数	描述	值
networking	集群网络的配置。	对象 **注意** 您不能在安装后修改 **networking** 对象指定的参数。
networking.networkType	要安装的集群网络供应商 Container Network Interface（CNI）插件。	**OpenShiftSDN** 或 **OVNKubernetes**。默认值为 **OpenShiftSDN**。
networking.clusterNetwork	pod 的 IP 地址块。 默认值为 **10.128.0.0/14**，主机前缀为 **/23**。 如果您指定多个 IP 地址块，则块不得互相重叠。	一个对象数组。例如： networking: clusterNetwork: - cidr: 10.128.0.0/14 hostPrefix: 23
networking.clusterNetwork.cidr	使用 **networking.clusterNetwork** 时需要此项。IP 地址块。 一个 IPv4 网络。	使用 CIDR 形式的 IP 地址块。IPv4 块的前缀长度介于 **0** 到 **32** 之间。
networking.clusterNetwork.hostPrefix	分配给每个单独节点的子网前缀长度。例如，如果 **hostPrefix** 设为 **23**，则每个节点从所给的 **cidr** 中分配一个 **/23** 子网。**hostPrefix** 值 **23** 提供 510（2^(32 - 23)- 2）个 pod IP 地址。	子网前缀。 默认值为 **23**。
networking.serviceNetwork	服务的 IP 地址块。默认值为 **172.30.0.0/16**。 OpenShift SDN 和 OVN-Kubernetes 网络供应商只支持服务网络的一个 IP 地址块。	CIDR 格式具有 IP 地址块的数组。例如： networking: serviceNetwork: - 172.30.0.0/16

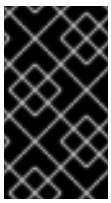
参数	描述	值
networking.machineNetwork	机器的 IP 地址块。 如果您指定多个 IP 地址块，则块不得互相重叠。	一个对象数组。例如： networking: machineNetwork: - cidr: 10.0.0.0/16
networking.machineNetwork.cidr	使用 **networking.machineNetwork** 时需要。IP 地址块。libvirt 以外的所有平台的默认值为 **10.0.0.0/16**。对于 libvirt，默认值为 **192.168.126.0/24**。	CIDR 表示法中的 IP 网络块。 例如： **10.0.0.0/16**。 **注意** 将 **networking.machineNetwork** 设置为与首选 NIC 所在的 CIDR 匹配。

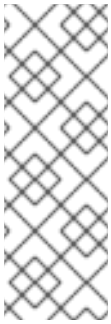
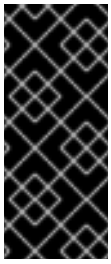
2.9.8.1.3. 可选配置参数

下表描述了可选安装配置参数：

表 2.28. 可选参数

参数	描述	值
additionalTrustBundle	添加到节点可信证书存储中的 PEM 编码 X.509 证书捆绑包。配置了代理时，也可以使用这个信任捆绑包。	字符串
compute	组成计算节点的机器的配置。	machine-pool 对象的数组。详情请查看以下"Machine-pool"表。
compute.architecture	决定池中机器的指令集合架构。目前不支持异构集群，因此所有池都必须指定相同的架构。有效值为 **amd64** （默认值）。	字符串

参数	描述	值
compute.hyperthreading	是否在计算机器上启用或禁用并发多线程或**超线程**。默认情况下，启用并发多线程以提高机器内核的性能。  **重要** 如果禁用并发多线程，请确保在容量规划时考虑到机器性能可能会显著降低的问题。	**Enabled** 或 **Disabled**
compute.name	使用 **compute** 时需要此值。机器池的名称。	**worker**
compute.platform	使用 **compute** 时需要此值。使用此参数指定托管 worker 机器的云供应商。此参数值必须与 **controlPlane.platform** 参数值匹配。	**aws**、**azure**、**gcp**、**openstack**、**ovirt**、**vsphere** 或 **{}**
compute.replicas	要置备的计算机器数量，也称为 worker 机器。	大于或等于 **2** 的正整数。默认值为 **3**。
controlPlane	组成 control plane 的机器的配置。	**MachinePool** 对象的数组。详情请查看以下"Machine-pool"表。
controlPlane.architecture	决定池中机器的指令集合架构。目前不支持异构集群，因此所有池都必须指定相同的架构。有效值为 **amd64** （默认值）。	字符串
controlPlane.hyperthreading	是否在 control plane 机器上启用或禁用并发多线程或**超线程**。默认情况下，启用并发多线程以提高机器内核的性能。  **重要** 如果禁用并发多线程，请确保在容量规划时考虑到机器性能可能会显著降低的问题。	**Enabled** 或 **Disabled**
controlPlane.name	使用 **controlPlane** 时需要。机器池的名称。	**master**

参数	描述	值
controlPlane.platform	使用 **controlPlane** 时需要。使用此参数指定托管 control plane 机器的云供应商。此参数值必须与 **compute.platform** 参数值匹配。	**aws**、**azure**、**gcp**、**openstack**、**ovirt**、**vsphere** 或 **{}**
controlPlane.replicas	要置备的 control plane 机器数量。	唯一支持的值是 **3**，它是默认值。
credentialsMode	Cloud Credential Operator（CCO）模式。如果没有指定任何模式，CCO 会动态地尝试决定提供的凭证的功能，在支持多个模式的平台上使用 mint 模式。  **注意** 不是所有 CCO 模式都支持所有云供应商。如需有关 CCO 模式的更多信息，请参阅*Red Hat Operator 参考指南*内容中的 *Cloud Credential Operator* 条目。	**Mint**、**Passthrough**、**Manual** 或空字符串(**""**)。
fips	启用或禁用 FIPS 模式。默认为 **false**（禁用）。如果启用了 FIPS 模式，运行 OpenShift Container Platform 的 Red Hat Enterprise Linux CoreOS (RHCOS) 机器会绕过默认的 Kubernetes 加密套件，并使用由 RHCOS 提供的加密模块。  **重要** 只有在 **x86_64** 架构中的 OpenShift Container Platform 部署支持 FIPS 验证的/Modules in Process 加密库。  **注意** 如果使用 Azure File 存储，则无法启用 FIPS 模式。	**false** 或 **true**

参数	描述	值
imageContentSources	release-image 内容的源和仓库。	对象数组。包括一个 **source** 以及可选的 **mirrors**，如下表所示。
imageContentSources.source	使用 **imageContentSources** 时需要。指定用户在镜像拉取规格中引用的仓库。	字符串
imageContentSources.mirrors	指定可能还包含同一镜像的一个或多个仓库。	字符串数组
publish	如何发布或公开集群的面向用户的端点，如 Kubernetes API、OpenShift 路由。	**Internal** 或 **External**。把 **publish** 设置为 **Internal** 以部署一个私有集群，它不能被互联网访问。默认值为 **External**。
sshKey	用于验证集群机器访问的 SSH 密钥或密钥。 **注意** 对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 **ssh-agent** 进程使用的 SSH 密钥。	一个或多个密钥。例如： sshKey: <key1> <key2> <key3>

2.9.8.1.4. 可选的 AWS 配置参数

下表描述了可选的 AWS 配置参数：

表 2.29. 可选的 AWS 参数

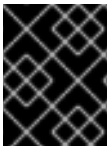
参数	描述	值
compute.platform.aws.amiID	用于为集群引导计算机器的 AWS AMI。对于需要自定义 RHCOS AMI 的区域来说，这是必需的。	属于集合 AWS 区域的任何已发布或自定义 RHCOS AMI。
compute.platform.aws.rootVolume.iops	为根卷保留的每秒输入/输出操作 (IOPS) 数。	整数，如 **4000**。

参数	描述	值
compute.platform.aws.rootVolume.size	以 GiB 为单位的根卷大小。	整数，如 **500**。
compute.platform.aws.rootVolume.type	根卷的类型。	有效的 AWS EBS 卷类型，如 **io1**。
compute.platform.aws.rootVolume.kmsKeyARN	KMS 密钥的 Amazon 资源名称（密钥 ARN）。这是使用特定 KMS 密钥加密 worker 节点的操作系统卷。	有效的 密钥 ID 或密钥 ARN。
compute.platform.aws.type	计算机器的 EC2 实例类型。	有效的 AWS 实例类型，如 **c5.9xlarge**。
compute.platform.aws.zones	安装程序在其中为计算机机器池创建机器的可用区。如果您提供自己的 VPC，则必须在那个可用域中提供一个子网。	有效 AWS 可用区的列表，如 **us-east-1c**，以 YAML 序列表示。
compute.aws.region	安装程序在其中创建计算资源的 AWS 区域。	任何有效的 AWS 区域，如 **us-east-1**。
controlPlane.platform.aws.amiID	用于为集群引导 control plane 机器的 AWS AMI。对于需要自定义 RHCOS AMI 的区域来说，这是必需的。	属于集合 AWS 区域的任何已发布或自定义 RHCOS AMI。
controlPlane.platform.aws.rootVolume.kmsKeyARN	KMS 密钥的 Amazon 资源名称（密钥 ARN）。这需要使用特定的 KMS 密钥加密 control plane 节点的操作系统卷。	有效的 密钥 ID 和密钥 ARN。
controlPlane.platform.aws.type	control plane 机器的 EC2 实例类型。	有效的 AWS 实例类型，如 **c5.9xlarge**。
controlPlane.platform.aws.zones	安装程序在其中为 control plane 机器池创建机器的可用区。	有效 AWS 可用区的列表，如 **us-east-1c**，以 YAML 序列表示。
controlPlane.aws.region	安装程序在其中创建 control plane 资源的 AWS 区域。	有效的 AWS 区域，如 **us-east-1**。
platform.aws.amiID	用于为集群引导所有机器的 AWS AMI。如果设置，AMI 必须属于与集群相同的区域。对于需要自定义 RHCOS AMI 的区域来说，这是必需的。	属于集合 AWS 区域的任何已发布或自定义 RHCOS AMI。

参数	描述	值
platform.aws.serviceEndpoints.name	AWS 服务端点名称。只有在必须使用替代 AWS 端点（如 FIPS）时，才需要自定义端点。可以为 EC2、S3、IAM、Elastic Load Balancing、Tagging、Route 53 和 STS AWS 服务指定自定义 API 端点。	有效的 AWS 服务端点名称。
platform.aws.serviceEndpoints.url	AWS 服务端点 URL。URL 必须使用 **https** 协议，主机必须信任该证书。	有效的 AWS 服务端点 URL。
platform.aws.userTags	键与值的映射，安装程序将其作为标签添加到它所创建的所有资源。	任何有效的 YAML 映射，如 **<key>: <value>** 格式的键值对。如需有关 AWS 标签的更多信息，请参阅 AWS 文档中的标记您的 Amazon EC2 资源。
platform.aws.subnets	如果您提供 VPC，而不是让安装程序为您创建 VPC，请指定要使用的集群子网。子网必须是您指定的同一 **machineNetwork[].cidr** 范围的一部分。对于标准集群，为每个可用区指定一个公共和私有子网。对于私有集群，为每个可用区指定一个私有子网。	有效的子网 ID。

2.9.8.2. AWS 的自定义 install-config.yaml 文件示例

您可以自定义 **install-config.yaml** 文件，以指定有关 OpenShift Container Platform 集群平台的更多信息，或修改所需参数的值。



重要

此示例 YAML 文件仅供参考。您必须使用安装程序来获取 **install-config.yaml** 文件，并且修改该文件。

```
apiVersion: v1
baseDomain: example.com (1)
credentialsMode: Mint (2)
controlPlane: (3) (4)
  hyperthreading: Enabled (5)
  name: master
  platform:
    aws:
      zones:
```

```
      - us-gov-west-1a
      - us-gov-west-1b
      rootVolume:
        iops: 4000
        size: 500
        type: io1 6
      type: m5.xlarge
  replicas: 3
compute: 7
- hyperthreading: Enabled 8
  name: worker
  platform:
    aws:
      rootVolume:
        iops: 2000
        size: 500
        type: io1 9
      type: c5.4xlarge
      zones:
      - us-gov-west-1c
  replicas: 3
metadata:
  name: test-cluster 10
networking:
  clusterNetwork:
  - cidr: 10.128.0.0/14
    hostPrefix: 23
  machineNetwork:
  - cidr: 10.0.0.0/16
  networkType: OpenShiftSDN
  serviceNetwork:
  - 172.30.0.0/16
platform:
  aws:
    region: us-gov-west-1
    userTags:
      adminContact: jdoe
      costCenter: 7536
    subnets: 11
    - subnet-1
    - subnet-2
    - subnet-3
    amiID: ami-96c6f8f7 12
    serviceEndpoints: 13
      - name: ec2
        url: https://vpce-id.ec2.us-west-2.vpce.amazonaws.com
    hostedZone: Z3URY6TWQ91KVV 14
fips: false 15
sshKey: ssh-ed25519 AAAA... 16
publish: Internal 17
pullSecret: '{"auths": ...}' 18
additionalTrustBundle: | 19
```

```
-----BEGIN CERTIFICATE-----
<MY_TRUSTED_CA_CERT>
-----END CERTIFICATE-----
```

1 10 18 必需。

2 可选：添加此参数来强制 Cloud Credential Operator（CCO）使用指定的模式，而不是让 CCO 动态尝试决定凭证的功能。如需有关 CCO 模式的详情，请参阅 *Red Hat Operator* 参考内容中的 *Cloud Credential Operator* 条目。

3 7 如果没有提供这些参数和值，安装程序会提供默认值。

4 **controlPlane** 部分是一个单映射，但 **compute** 部分是一系列映射。为满足不同数据结构的要求，**compute** 部分的第一行必须以连字符 **-** 开头，**controlPlane** 部分的第一行则不可以连字符开头。只使用一个 control plane 池。

5 8 是否要启用或禁用并发多线程或**超线程**。默认情况下，启用并发多线程以提高机器内核的性能。您可以通过将参数值设为 **Disabled** 来禁用。如果您在某些集群机器上禁用并发多线程，则必须在所有集群机器上禁用。

重要

如果禁用并发多线程，请确保在容量规划时考虑到机器性能可能会显著降低的问题。如果您对机器禁用并发多线程，请使用较大的实例类型，如 **m4.2xlarge** 或 **m5.2xlarge**。

6 9 要为 etcd 配置更快的存储，特别是对于较大的集群，请将存储类型设置为 **io1**，并将 **iops** 设为 **2000**。

11 如果您提供自己的 VPC，为集群使用的每个可用区指定子网。

12 用于为集群引导机器的 AMI ID。如果设置，AMI 必须属于与集群相同的区域。

13 AWS 服务端点。在安装到未知 AWS 区域时，需要自定义端点。端点 URL 必须使用 **https** 协议，主机必须信任该证书。

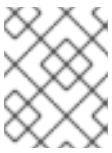
14 您现有 Route 53 私有托管区的 ID。提供现有的托管区需要您提供自己的 VPC，托管区已在安装集群前与 VPC 关联。如果未定义，安装程序会创建一个新的托管区。

15 是否启用或禁用 FIPS 模式。默认情况下不启用 FIPS 模式。如果启用了 FIPS 模式，运行 OpenShift Container Platform 的 Red Hat Enterprise Linux CoreOS (RHCOS) 机器会绕过默认的 Kubernetes 加密套件，并使用由 RHCOS 提供的加密模块。

重要

只有在 **x86_64** 架构中的 OpenShift Container Platform 部署支持 FIPS 验证的/Modules in Process 加密库。

16 您可以选择提供您用来访问集群中机器的 **sshKey** 值。



注意

对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 **ssh-agent** 进程使用的 SSH 密钥。

17 如何发布集群的面向用户的端点。把 **publish** 设置为 **Internal** 以部署一个私有集群，它不能被互联网访问。默认值为 **External**。

19 自定义 CA 证书。当部署到 AWS C2S Secret 区域时，这是必需的，因为 AWS API 需要自定义 CA 信任捆绑包。

2.9.8.3. 没有公布的 RHCOS AMI 的 AWS 区域

您可以将 OpenShift Container Platform 集群部署到 Amazon Web Services（AWS）区域，而无需对 Red Hat Enterprise Linux CoreOS（RHCOS）Amazon Machine Image（AMI）或 AWS 软件开发 kit（SDK）的原生支持。如果 AWS 区域没有可用的已公布的 AMI，您可以在安装集群前上传自定义 AMI。如果您要将集群部署到 AWS 政府区域，则需要此参数。

如果您部署到没有公布的 RHCOS AMI 的非机构区域，且您没有指定自定义的 AMI，安装程序会自动将 **us-east-1** AMI 复制到用户帐户。然后，安装程序使用默认或用户指定的密钥管理服务（KMS）密钥创建带有加密 EBS 卷的 control plane 机器。这允许 AMI 跟踪与公布的 RHCOS AMI 相同的进程工作流。

在集群创建过程中，无法从终端中选择没有原生支持 RHCOS AMI 的区域，因为它没有发布。但是，您可以通过在 **install-config.yaml** 文件中配置自定义 AMI 来安装到这个区域。

2.9.8.4. 在 AWS 中上传自定义 RHCOS AMI

如果要部署到自定义 Amazon Web Services（AWS）区域，您必须上传属于该区域的自定义 Red Hat Enterprise Linux CoreOS（RHCOS）Amazon Machine Image（AMI）。

先决条件

- 已配置了一个 AWS 帐户。
- 已使用所需的 IAM 服务角色创建 Amazon S3 存储桶。
- 将 RHCOS VMDK 文件上传到 Amazon S3。RHCOS VMDK 文件必须是小于或等于您要安装的 OpenShift Container Platform 版本的最高版本。
- 您下载了 AWS CLI 并安装到您的计算机上。请参阅使用捆绑安装程序安装 AWS CLI。

流程

1. 将 AWS 配置集导出为环境变量：

```
$ export AWS_PROFILE=<aws_profile> 1
```

1 拥有 AWS 凭证的 AWS 配置集名称，如 **govcloud**。

2. 将与自定义 AMI 关联的区域导出为环境变量：

```
$ export AWS_DEFAULT_REGION=<aws_region> 1
```

1 AWS 区域，如 **us-gov-east-1**。

3. 将上传至 Amazon S3 的 RHCOS 版本导出为环境变量：

```
$ export RHCOS_VERSION=<version> 1
```

1 RHCOS VMDK 版本，如 **4.6.0**。

4. 将 Amazon S3 存储桶名称导出为环境变量：

```
$ export VMIMPORT_BUCKET_NAME=<s3_bucket_name>
```

5. 创建 **containers.json** 文件并定义 RHCOS VMDK 文件：

```
$ cat <<EOF > containers.json
{
  "Description": "rhcos-${RHCOS_VERSION}-x86_64-aws.x86_64",
  "Format": "vmdk",
  "UserBucket": {
    "S3Bucket": "${VMIMPORT_BUCKET_NAME}",
    "S3Key": "rhcos-${RHCOS_VERSION}-x86_64-aws.x86_64.vmdk"
  }
}
EOF
```

6. 将 RHCOS 磁盘导入为 Amazon EBS 快照：

```
$ aws ec2 import-snapshot --region ${AWS_DEFAULT_REGION} \
    --description "<description>" \ 1
    --disk-container "file://<file_path>/containers.json" 2
```

1 导入 RHCOS 磁盘的描述，如 **rhcos-${RHCOS_VERSION}-x86_64-aws.x86_64**。

2 描述 RHCOS 磁盘的 JSON 文件的文件路径。JSON 文件应包含您的 Amazon S3 存储桶名称和密钥。

7. 检查镜像导入的状态：

```
$ watch -n 5 aws ec2 describe-import-snapshot-tasks --region ${AWS_DEFAULT_REGION}
```

输出示例

```
{
    "ImportSnapshotTasks": [
        {
            "Description": "rhcos-4.6.0-x86_64-aws.x86_64",
            "ImportTaskId": "import-snap-fh6i8uil",
            "SnapshotTaskDetail": {
                "Description": "rhcos-4.6.0-x86_64-aws.x86_64",
                "DiskImageSize": 819056640.0,
                "Format": "VMDK",
                "SnapshotId": "snap-06331325870076318",
                "Status": "completed",
                "UserBucket": {
                    "S3Bucket": "external-images",
                    "S3Key": "rhcos-4.6.0-x86_64-aws.x86_64.vmdk"
```

```
            }
          }
        }
    ]
}
```

复制 **SnapshotId** 以注册镜像。

8. 从 RHCOS 快照创建自定义 RHCOS AMI:

```
$ aws ec2 register-image \
  --region ${AWS_DEFAULT_REGION} \
  --architecture x86_64 \ 1
  --description "rhcos-${RHCOS_VERSION}-x86_64-aws.x86_64" \ 2
  --ena-support \
  --name "rhcos-${RHCOS_VERSION}-x86_64-aws.x86_64" \ 3
  --virtualization-type hvm \
  --root-device-name '/dev/xvda' \
  --block-device-mappings 'DeviceName=/dev/xvda,Ebs=
{DeleteOnTermination=true,SnapshotId=<snapshot_ID>}' 4
```

1. RHCOS VMDK 架构类型，如 **x86_64**、**s390x** 或 **ppc64le**。
2. 来自导入快照的 **Description**。
3. RHCOS AMI 的名称。
4. 导入的快照中的 **SnapshotID**。

如需了解更多有关这些 API 的信息，请参阅 AWS 文档 导入快照 和 创建由 EBS 支持的 AMI。

2.9.8.5. 在安装过程中配置集群范围代理

生产环境可能会拒绝直接访问互联网，而是提供 HTTP 或 HTTPS 代理。您可以通过在 **install-config.yaml** 文件中配置代理设置，将新的 OpenShift Container Platform 集群配置为使用代理。

先决条件

- 您有一个现有的 **install-config.yaml** 文件。
- 您检查了集群需要访问的站点，并决定是否需要绕过代理。默认情况下代理所有集群出口流量，包括对托管云供应商 API 的调用。您需要将站点添加到 **Proxy** 对象的 **spec.noProxy** 字段来绕过代理。

注意

Proxy 对象 **status.noProxy** 字段使用安装配置中的 **networking.machineNetwork[].cidr**、**networking.clusterNetwork[].cidr** 和 **networking.serviceNetwork[]** 字段的值填充。

对于在 Amazon Web Services(AWS)、Google Cloud Platform(GCP)、Microsoft Azure 和 Red Hat OpenStack Platform(RHOSP)上安装, **Proxy** 对象 **status.noProxy** 字段也会使用实例元数据端点填充(**169.254.169.254**)。

- 如果您的集群位于 AWS 上，请将 **ec2.<region>.amazonaws.com**、**elasticloadbalancing.<region>.amazonaws.com** 和 **s3.<region>.amazonaws.com** 端点添加到 VPC 端点。需要这些端点才能完成节点到 AWS EC2 API 的请求。由于代理在容器级别而不是节点级别工作，因此您必须通过 AWS 专用网络将这些请求路由到 AWS EC2 API。在代理服务器中的允许列表中添加 EC2 API 的公共 IP 地址是不够的。

流程

1. 编辑 **install-config.yaml** 文件并添加代理设置。例如：

```
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: example.com 3
additionalTrustBundle: | 4
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
...
```

1. 用于创建集群外 HTTP 连接的代理 URL。URL 必须是 **http**。
2. 用于创建集群外 HTTPS 连接的代理 URL。
3. 要排除在代理中的目标域名、IP 地址或其他网络 CIDR 的逗号分隔列表。在域前面加 **.** 来仅匹配子域。例如：**.y.com** 匹配 **x.y.com**，但不匹配 **y.com**。使用 ***** 绕过所有目的地的代理。
4. 如果提供，安装程序会在 **openshift-config 命名空间中生成名为 user-ca- bundle** 的配置映射来保存额外的 CA 证书。如果您提供 **additionalTrustBundle** 和至少一个代理设置，则 **Proxy** 对象会被配置为引用 **trustedCA** 字段中的 **user-ca-bundle** 配置映射。然后，Cluster Network Operator 会创建一个 **trusted-ca-bundle** 配置映射，该配置映射将为 **trustedCA** 参数指定的内容与 RHCOS 信任捆绑包合并。**additionalTrustBundle** 字段是必需的，除非代理的身份证书由来自 RHCOS 信任捆绑包的颁发机构签名。

注意

安装程序不支持代理的 **readinessEndpoints** 字段。

2. 保存该文件，并在安装 OpenShift Container Platform 时引用。

安装程序会创建一个名为 **cluster** 的集群范围代理，该代理使用提供的 **install-config.yaml** 文件中的代理设置。如果没有提供代理设置，仍然会创建一个 **cluster Proxy** 对象，但它会有一个空 **spec**。

注意

只支持名为 **cluster** 的 **Proxy** 对象，且无法创建额外的代理。

2.9.9. 部署集群

您可以在兼容云平台中安装 OpenShift Container Platform。

重要

安装程序的 **create cluster** 命令只能在初始安装过程中运行一次。

先决条件

- 配置托管集群的云平台的帐户。
- 获取 OpenShift Container Platform 安装程序以及集群的 pull secret。

流程

1. 更改为包含安装程序的目录并初始化集群部署：

```
$ ./openshift-install create cluster --dir <installation_directory> \ 1
    --log-level=info 2
```

1 对于 **<installation_directory>**，请指定自定义 **./install-config.yaml** 文件的位置。

2 要查看不同的安装详情，请指定 **warn**、**debug** 或 **error**，而不要指定 **info**。

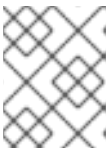
注意

如果您在主机上配置的云供应商帐户没有足够的权限来部署集群，安装过程将会停止，并且显示缺少的权限。

集群部署完成后，终端会显示访问集群的信息，包括指向其 Web 控制台的链接和 **kubeadmin** 用户的凭证。

输出示例

```
...
INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run 'export
KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'
INFO Access the OpenShift web-console here: https://console-openshift-
console.apps.mycluster.example.com
INFO Login to the console with user: "kubeadmin", and password: "4vYBz-Ee6gm-ymBZj-
Wt5AL"
INFO Time elapsed: 36m22s
```



注意

当安装成功时，集群访问和凭证信息还会输出到 **<installation_directory>/.openshift_install.log**。

重要

- 安装程序生成的 Ignition 配置文件包含在 24 小时后过期的证书，然后在过期时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外情况是，您需要手动批准待处理的 **node-bootstrapper** 证书签名请求（CSR）来恢复 kubelet 证书。如需更多信息，请参阅*从过期的 control plane 证书中恢复*的文档。
- 建议您在生成 12 小时后使用 Ignition 配置文件，因为集群安装后 24 小时证书从 16 小时轮转至 22 小时。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中运行证书更新时避免安装失败。

重要

您不得删除安装程序或安装程序所创建的文件。需要这两者才能删除集群。

2. 可选：从您用来安装集群的 IAM 帐户删除或禁用 **AdministratorAccess** 策略。

注意

只有在安装过程中才需要 **AdministratorAccess** 策略提供的升级权限。

2.9.10. 通过下载二进制文件安装 OpenShift CLI

您需要安装 CLI（**oc**） 来使用命令行界面与 OpenShift Container Platform 进行交互。您可在 Linux、Windows 或 macOS 上安装 **oc**。

重要

如果安装了旧版本的 **oc**，则无法使用 OpenShift Container Platform 4.6 中的所有命令。下载并安装新版本的 **oc**。

2.9.10.1. 在 Linux 上安装 OpenShift CLI

您可以按照以下流程在 Linux 上安装 OpenShift CLI（**oc**）二进制文件。

流程

1. 进入到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 在 **Version** 下拉菜单中选择相应的版本。
3. 单击 **OpenShift v4.6 Linux 客户端** 条目旁边的 **Download Now**，再保存文件。
4. 解包存档：

```
$ tar xvzf <file>
```

5. 把 **oc** 二进制代码放到 **PATH** 中的目录中。
执行以下命令可以查看当前的 **PATH** 设置：

```
$ echo $PATH
```

安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
$ oc <command>
```

2.9.10.2. 在 Windows 上安装 OpenShift CLI

您可以按照以下流程在 Windows 上安装 OpenShift CLI（**oc**）二进制代码。

流程

1. 进入到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 在 **Version** 下拉菜单中选择相应的版本。
3. 单击 **OpenShift v4.6 Windows 客户端**条目旁边的 **Download Now**，再保存文件。
4. 使用 ZIP 程序解压存档。
5. 把 **oc** 二进制代码放到 **PATH** 中的目录中。
   要查看您的 **PATH**，请打开命令提示窗口并执行以下命令：

   ```
   C:\> path
   ```

安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
C:\> oc <command>
```

2.9.10.3. 在 macOS 上安装 OpenShift CLI

您可以按照以下流程在 macOS 上安装 OpenShift CLI（**oc**）二进制代码。

流程

1. 进入到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 在 **Version** 下拉菜单中选择相应的版本。
3. 单击 **OpenShift v4.6 MacOSX 客户端**条目旁边的 **Download Now**，再保存文件。
4. 解包和解压存档。
5. 将 **oc** 二进制文件移到 PATH 的目录中。
   要查看您的 **PATH**，打开一个终端窗口并执行以下命令：

   ```
   $ echo $PATH
   ```

安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
$ oc <command>
```

2.9.11. 使用 CLI 登录到集群

您可以通过导出集群 **kubeconfig** 文件，以默认系统用户身份登录集群。**kubeconfig** 文件包含关于集群的信息，供 CLI 用于将客户端连接到正确集群和 API 服务器。该文件特只适用于一个特定的集群，在 OpenShift Container Platform 安装过程中创建。

先决条件

- 已部署了 OpenShift Container Platform 集群。
- 已安装 **oc** CLI。

流程

1. 导出 **kubeadmin** 凭证：

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

2. 使用导出的配置，验证能否成功运行 **oc** 命令：

```
$ oc whoami
```

输出示例

```
system:admin
```

2.9.12. 使用 Web 控制台登录到集群

kubeadmin 用户默认在 OpenShift Container Platform 安装后存在。您可以使用 OpenShift Container Platform Web 控制台以 **kubeadmin** 用户身份登录集群。

先决条件

- 有访问安装主机的访问权限。
- 您完成了集群安装，所有集群 Operator 都可用。

流程

1. 从安装主机上的 **kubeadmin -password** 文件中获取 kubeadmin 用户的密码：

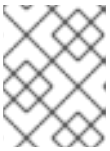
```
$ cat <installation_directory>/auth/kubeadmin-password
```

注意

另外，您还可以从安装主机上的 **<installation_directory>/.openshift_install.log** 日志文件获取 **kubeadmin** 密码。

2. 列出 OpenShift Container Platform Web 控制台路由：

```
$ oc get routes -n openshift-console | grep 'console-openshift'
```



注意

另外，您还可以从安装主机上的 **<installation_directory>/.openshift_install.log 日志** 文件获取 OpenShift Container Platform 路由。

输出示例

```
console     console-openshift-console.apps.<cluster_name>.<base_domain>          console
https   reencrypt/Redirect   None
```

3. 在 Web 浏览器中导航到上一命令输出中包括的路由，以 **kubeadmin** 用户身份登录。

其他资源

- 如需有关访问和了解 OpenShift Container Platform Web 控制台的更多信息，请参阅访问 Web 控制台。

2.9.13. OpenShift Container Platform 的 Telemetry 访问

在 OpenShift Container Platform 4.6 中，默认运行的 Telemetry 服务提供有关集群健康状况和成功更新的指标，需要访问互联网。如果您的集群连接到互联网，Telemetry 会自动运行，而且集群会注册到 OpenShift Cluster Manager。

确认 OpenShift Cluster Manager 清单正确后，可以由 Telemetry 自动维护，也可以使用 OpenShift Cluster Manager 手动维护，使用订阅监控来跟踪帐户或多集群级别的 OpenShift Container Platform 订阅。

其他资源

- 有关 Telemetry 服务的更多信息，请参阅关于远程健康监控。

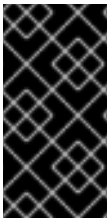
2.9.14. 后续步骤

- 验证安装。
- 自定义集群。
- 如果需要，您可以选择不使用远程健康报告。
- 如果需要，您可以删除云供应商凭证。

2.10. 使用 CLOUDFORMATION 模板在 AWS 中用户置备的基础架构上安装集群

在 OpenShift Container Platform 版本 4.6 中，您可以使用您提供的基础架构在 Amazon Web Services（AWS）上安装集群。

创建此基础架构的一种方法是使用提供的 CloudFormation 模板。您可以修改模板来自定义基础架构，或使用其包含的信息来按照公司策略创建 AWS 对象。



重要

进行用户置备的基础架构安装的步骤仅作为示例。使用您提供的基础架构安装集群需要了解云供应商和 OpenShift Container Platform 安装过程。提供的几个 CloudFormation 模板可帮助完成这些步骤，或者帮助您自行建模。您也可以自由选择通过其他方法创建所需的资源；模板仅作示例之用。

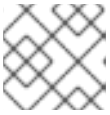
2.10.1. 先决条件

- 您可以参阅有关 OpenShift Container Platform 安装和更新流程的详细信息。
- 已将 AWS 帐户配置为托管集群。

重要

如果您的计算机上存储有 AWS 配置集，则不要在使用多因素验证设备的同时使用您生成的临时会话令牌。在集群的整个生命周期中，集群会持续使用您的当前 AWS 凭证来创建 AWS 资源，因此您必须使用基于密钥的长期凭证。要生成适当的密钥，请参阅 AWS 文档中的管理 IAM 用户的访问密钥。您可在运行安装程序时提供密钥。

- 您下载了 AWS CLI 并安装到您的计算机上。请参阅 AWS 文档中的使用捆绑安装程序（Linux、macOS 或 Unix）安装 AWS CLI。
- 如果使用防火墙，则必须将其配置为允许集群需要访问的站点。



注意

如果您要配置代理，请务必也要查看此站点列表。

- 如果不允许系统管理身份和访问管理（IAM），集群管理员可以 手动创建和维护 IAM 凭证。手动模式也可以用于云 IAM API 无法访问的环境中。

2.10.2. OpenShift Container Platform 的互联网访问

在 OpenShift Container Platform 4.6 中，您需要访问互联网来安装集群。

您必须具有以下互联网访问权限：

- 访问 OpenShift Cluster Manager 以下载安装程序并执行订阅管理。如果集群可以访问互联网，并且没有禁用 Telemetry，该服务会自动授权您的集群。
- 访问 Quay.io，以获取安装集群所需的软件包。
- 获取执行集群更新所需的软件包。

重要

如果您的集群无法直接访问互联网，则可以在置备的某些类基础架构上执行受限网络安装。在此过程中，您要下载所需的内容，并使用它在镜像 registry（mirror registry） 中填充安装集群并生成安装程序所需的软件包。对于某些安装类型，集群要安装到的环境不需要访问互联网。在更新集群之前，要更新 registry 镜像系统中的内容。

2.10.3. 所需的 AWS 基础架构组件

要在 Amazon Web Services (AWS) 中用户置备的基础架构上安装 OpenShift Container Platform，您必须手动创建机器及其支持的基础架构。

如需有关不同平台集成测试的更多信息，请参阅 OpenShift Container Platform 4.x Tested Integrations 页面。

通过使用提供的 CloudFormation 模板，您可以创建代表以下组件的 AWS 资源堆栈：

- 一个 AWS Virtual Private Cloud (VPC)
- 网络和负载均衡组件
- 安全组和角色
- 一个 OpenShift Container Platform bootstrap 节点
- OpenShift Container Platform control plane 节点
- 一个 OpenShift Container Platform 计算节点

或者，您可以手动创建组件，也可以重复使用满足集群要求的现有基础架构。查看 CloudFormation 模板，了解组件如何相互连接的更多详情。

2.10.3.1. 集群机器

以下机器需要 **AWS::EC2::Instance** 对象：

- bootstrap 机器。安装过程中需要此机器，但可在集群部署后删除。
- 三个 control plane 机器。control plane 机器不受机器集的管控。
- 计算机器。在安装过程中创建至少两台计算（compute）机器（也称为 worker 机器）。这些机器不受机器集的管控。

您可以通过提供的 CloudFormation 模板，为集群机器使用以下实例类型。

重要

如果您的区域中没有 **m4** 实例类型，例如 **eu-west-3**，请改为使用 **m5** 类型。

表 2.30. 机器的实例类型

实例类型	bootstrap	Control plane	Compute
i3.large	x		
m4.large			x
m4.xlarge		x	x
m4.2xlarge		x	x

实例类型	bootstrap	Control plane	Compute
m4.4xlarge		x	x
m4.8xlarge		x	x
m4.10xlarge		x	x
m4.16xlarge		x	x
m5.large			x
m5.xlarge		x	x
m5.2xlarge		x	x
m5.4xlarge		x	x
m5.8xlarge		x	x
m5.10xlarge		x	x
m5.16xlarge		x	x
m6i.xlarge		x	x
c4.2xlarge		x	x
c4.4xlarge		x	x
c4.8xlarge		x	x
r4.large			x
r4.xlarge		x	x
r4.2xlarge		x	x
r4.4xlarge		x	x
r4.8xlarge		x	x
r4.16xlarge		x	x

您可能能够使用符合这些实例类型规格的其他实例类型。

2.10.3.2. 其他基础架构组件

- VPC
- DNS 条目
- 负载均衡器（典型或网络）和监听器
- 公共和专用路由 53 区域
- 安全组
- IAM 角色
- S3 存储桶

如果您在断开连接的环境或使用代理的环境中工作，则无法访问 EC2 和 ELB 端点的公共 IP 地址。要访问这些端点，您必须创建一个 VPC 端点，并将其附加到集群使用的子网。创建以下端点：

- **ec2.<region>.amazonaws.com**
- **elasticloadbalancing.<region>.amazonaws.com**
- **s3.<region>.amazonaws.com**

所需的 VPC 组件

您必须提供合适的 VPC 和子网，以便与您的机器通信。

组件	AWS 类型	描述
VPC	• **AWS::EC2::VPC** • **AWS::EC2::VPCEndpoint**	您必须提供一个公共 VPC 供集群使用。VPC 使用引用每个子网的路由表的端点，以改进与托管在 S3 中的 registry 的通信。
公共子网	• **AWS::EC2::Subnet** • **AWS::EC2::SubnetNetworkAclAssociation**	您的 VPC 必须有 1 到 3 个可用区的公共子网，并将其与适当的入口规则关联。
互联网网关	• **AWS::EC2::InternetGateway** • **AWS::EC2::VPCGatewayAttachment** • **AWS::EC2::RouteTable** • **AWS::EC2::Route** • **AWS::EC2::SubnetRouteTableAssociation** • **AWS::EC2::NatGateway** • **AWS::EC2::EIP**	您必须有一个公共互联网网关，以及附加到 VPC 的公共路由。在提供的模板中，每个公共子网都有一个具有 EIP 地址的 NAT 网关。这些 NAT 网关允许集群资源（如专用子网实例）访问互联网，而有些受限网络或代理场景则不需要它们。

<table>
<tr><th>组件</th><th>AWS 类型</th><th colspan="2">描述</th></tr>
<tr><td rowspan="7">网络访问控制</td><td rowspan="7">AWS::EC2::NetworkAcl
AWS::EC2::NetworkAclEntry</td><td colspan="2">您必须允许 VPC 访问下列端口：</td></tr>
<tr><td>端口</td><td>原因</td></tr>
<tr><td>80</td><td>入站 HTTP 流量</td></tr>
<tr><td>443</td><td>入站 HTTPS 流量</td></tr>
<tr><td>22</td><td>入站 SSH 流量</td></tr>
<tr><td>1024 - 65535</td><td>入站临时流量</td></tr>
<tr><td>0 - 65535</td><td>出站临时流量</td></tr>
<tr><td>专用子网</td><td>AWS::EC2::Subnet
AWS::EC2::RouteTable
AWS::EC2::SubnetRouteTableAssociation</td><td colspan="2">您的 VPC 可以具有私有子网。提供的 CloudFormation 模板可为 1 到 3 个可用区创建专用子网。如果您使用专用子网，必须为其提供适当的路由和表。</td></tr>
</table>

所需的 DNS 和负载均衡组件

您的 DNS 和负载均衡器配置需要使用公共托管区，并可使用类似安装程序使用的专用托管区（如果安装程序置备了集群的基础架构）。您必须创建一个解析到负载均衡器的 DNS 条目。**api.<cluster_name>.<domain>** 的条目必须指向外部负载均衡器，**api-int.<cluster_name>.<domain>** 的条目则必须指向内部负载均衡器。

集群还需要负载均衡器，以及监听端口 6443（用于 Kubernetes API 及其扩展）和端口 22623（用于新机器的 Ignition 配置文件）的监听程序。目标是 control plane 节点（也称为 master 节点）。集群外的客户端和集群内的节点都必须能够访问端口 6443。集群内的节点必须能够访问端口 22623。

组件	AWS 类型	描述
DNS	**AWS::Route53::HostedZone**	内部 DNS 的托管区。
etcd 记录集	**AWS::Route53::RecordSet**	control plane 机器的 etcd 注册记录。
公共负载均衡器	**AWS::ElasticLoadBalancingV2::LoadBalancer**	公共子网的负载均衡器。

组件	AWS 类型	描述
外部 API 服务器记录	**AWS::Route53::RecordSetGroup**	外部 API 服务器的别名记录。
外部监听程序	**AWS::ElasticLoadBalancingV2::Listener**	为外部负载均衡器监听端口 6443 的监听程序。
外部目标组	**AWS::ElasticLoadBalancingV2::TargetGroup**	外部负载均衡器的目标组。
专用负载均衡器	**AWS::ElasticLoadBalancingV2::LoadBalancer**	专用子网的负载均衡器。
内部 API 服务器记录	**AWS::Route53::RecordSetGroup**	内部 API 服务器的别名记录。
内部监听程序	**AWS::ElasticLoadBalancingV2::Listener**	为内部负载均衡器监听端口 22623 的监听程序。
内部目标组	**AWS::ElasticLoadBalancingV2::TargetGroup**	内部负载均衡器的目标组。
内部监听程序	**AWS::ElasticLoadBalancingV2::Listener**	为内部负载均衡器监听端口 6443 的监听程序。
内部目标组	**AWS::ElasticLoadBalancingV2::TargetGroup**	内部负载均衡器的目标组。

安全组

control plane 和 worker 机器需要访问下列端口：

组	类型	IP 协议	端口范围
MasterSecurityGroup	**AWS::EC2::SecurityGroup**	**icmp**	**0**
		tcp	**22**
		tcp	**6443**
		tcp	**22623**
WorkerSecurityGroup	**AWS::EC2::SecurityGroup**	**icmp**	**0**
		tcp	**22**
BootstrapSecurityGroup	**AWS::EC2::SecurityGroup**	**tcp**	**22**
		tcp	**19531**

control plane 入口

control plane 机器需要以下入口组。每个入口组都是 **AWS::EC2::SecurityGroupIngress** 资源。

入口组	描述	IP 协议	端口范围
MasterIngressEtcd	etcd	**tcp**	**2379- 2380**
MasterIngressVxlan	Vxlan 数据包	**udp**	**4789**
MasterIngressWorkerVxlan	Vxlan 数据包	**udp**	**4789**
MasterIngressInternal	内部集群通信和 Kubernetes 代理指标	**tcp**	**9000 - 9999**
MasterIngressWorkerInternal	内部集群通信	**tcp**	**9000 - 9999**
MasterIngressKube	kubernetes kubelet、调度程序和控制器管理器	**tcp**	**10250 - 10259**
MasterIngressWorkerKube	kubernetes kubelet、调度程序和控制器管理器	**tcp**	**10250 - 10259**

入口组	描述	IP 协议	端口范围
MasterIngressIngressServices	Kubernetes 入口服务	**tcp**	**30000 - 32767**
MasterIngressWorkerIngressServices	Kubernetes 入口服务	**tcp**	**30000 - 32767**
MasterIngressGeneve	Geneve 数据包	**udp**	**6081**
MasterIngressWorkerGeneve	Geneve 数据包	**udp**	**6081**
MasterIngressIpsecIke	IPsec IKE 数据包	**udp**	**500**
MasterIngressWorkerIpsecIke	IPsec IKE 数据包	**udp**	**500**
MasterIngressIpsecNat	IPsec NAT-T 数据包	**udp**	**4500**
MasterIngressWorkerIpsecNat	IPsec NAT-T 数据包	**udp**	**4500**
MasterIngressIpsecEsp	IPsec ESP 数据包	**50**	**All**
MasterIngressWorkerIpsecEsp	IPsec ESP 数据包	**50**	**All**
MasterIngressInternalUDP	内部集群通信	**udp**	**9000 - 9999**
MasterIngressWorkerInternalUDP	内部集群通信	**udp**	**9000 - 9999**
MasterIngressIngressServicesUDP	Kubernetes 入口服务	**udp**	**30000 - 32767**

入口组	描述	IP 协议	端口范围
MasterIngress WorkerIngress ServicesUDP	Kubernetes 入口服务	**udp**	**30000 - 32767**

worker 入口

worker 机器需要以下入口组。每个入口组都是 **AWS::EC2::SecurityGroupIngress** 资源。

入口组	描述	IP 协议	端口范围
WorkerIngress Vxlan	Vxlan 数据包	**udp**	**4789**
WorkerIngress WorkerVxlan	Vxlan 数据包	**udp**	**4789**
WorkerIngress Internal	内部集群通信	**tcp**	**9000 - 9999**
WorkerIngress WorkerInternal	内部集群通信	**tcp**	**9000 - 9999**
WorkerIngress Kube	Kubernetes kubelet、调度程序和控制器管理器	**tcp**	**10250**
WorkerIngress WorkerKube	Kubernetes kubelet、调度程序和控制器管理器	**tcp**	**10250**
WorkerIngress IngressServices	Kubernetes 入口服务	**tcp**	**30000 - 32767**
WorkerIngress WorkerIngress Services	Kubernetes 入口服务	**tcp**	**30000 - 32767**
WorkerIngress Geneve	Geneve 数据包	**udp**	**6081**
WorkerIngress MasterGeneve	Geneve 数据包	**udp**	**6081**
WorkerIngress IpsecIke	IPsec IKE 数据包	**udp**	**500**

入口组	描述	IP 协议	端口范围
WorkerIngressMasterIpsecIke	IPsec IKE 数据包	**udp**	**500**
WorkerIngressIpsecNat	IPsec NAT-T 数据包	**udp**	**4500**
WorkerIngressMasterIpsecNat	IPsec NAT-T 数据包	**udp**	**4500**
WorkerIngressIpsecEsp	IPsec ESP 数据包	**50**	**All**
WorkerIngressMasterIpsecEsp	IPsec ESP 数据包	**50**	**All**
WorkerIngressInternalUDP	内部集群通信	**udp**	**9000 - 9999**
WorkerIngressMasterInternalUDP	内部集群通信	**udp**	**9000 - 9999**
WorkerIngressIngressServicesUDP	Kubernetes 入口服务	**udp**	**30000 - 32767**
WorkerIngressMasterIngressServicesUDP	Kubernetes 入口服务	**udp**	**30000 - 32767**

角色和实例配置集

您必须在 AWS 中为机器授予权限。提供的 CloudFormation 模板为以下 **AWS::IAM::Role** 对象授予机器 **Allow** 权限，并为每一组角色提供一个 **AWS::IAM::InstanceProfile**。如果不使用模板，您可以为机器授予以下宽泛权限或单独权限。

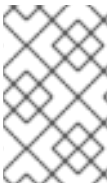
角色	影响	操作	资源
Master	**Allow**	**ec2:***	*
	Allow	**elasticloadbalancing:***	*

角色	影响	操作	资源
	Allow	**iam:PassRole**	*
	Allow	**s3:GetObject**	*
Worker	**Allow**	**ec2:Describe***	*
bootstrap	**Allow**	**ec2:Describe***	*
	Allow	**ec2:AttachVolume**	*
	Allow	**ec2:DetachVolume**	*

2.10.3.3. 证书签名请求管理

在使用您置备的基础架构时，集群只能有限地访问自动机器管理，因此您必须提供一种在安装后批准集群证书签名请求 (CSR) 的机制。**kube-controller-manager** 只能批准 kubelet 客户端 CSR。 **machine-approver** 无法保证使用 kubelet 凭证请求的提供证书的有效性，因为它不能确认是正确的机器发出了该请求。您必须决定并实施一种方法，以验证 kubelet 提供证书请求的有效性并进行批准。

2.10.3.4. 所需的 AWS 权限



注意

您的 IAM 用户必须具有 region **us-east-1** s 中的 permissions **tag:GetResources** 才能删除基本集群资源。作为 AWS API 要求的一部分，OpenShift Container Platform 安装程序在此区域中执行各种操作。

将 **AdministratorAccess** 策略附加到您在 Amazon Web Services (AWS) 中创建的 IAM 用户时，授予该用户所有需要的权限。要部署 OpenShift Container Platform 集群的所有组件，IAM 用户需要以下权限：

例 2.13. 安装所需的 EC2 权限

- **tag:TagResources**
- **tag:UntagResources**
- **ec2:AllocateAddress**
- **ec2:AssociateAddress**
- **ec2:AuthorizeSecurityGroupEgress**
- **ec2:AuthorizeSecurityGroupIngress**
- **ec2:CopyImage**
- **ec2:CreateNetworkInterface**
- **ec2:AttachNetworkInterface**

- **ec2:CreateSecurityGroup**
- **ec2:CreateTags**
- **ec2:CreateVolume**
- **ec2:DeleteSecurityGroup**
- **ec2:DeleteSnapshot**
- **ec2:DeleteTags**
- **ec2:DeregisterImage**
- **ec2:DescribeAccountAttributes**
- **ec2:DescribeAddresses**
- **ec2:DescribeAvailabilityZones**
- **ec2:DescribeDhcpOptions**
- **ec2:DescribeImages**
- **ec2:DescribeInstanceAttribute**
- **ec2:DescribeInstanceCreditSpecifications**
- **ec2:DescribeInstances**
- **ec2:DescribeInternetGateways**
- **ec2:DescribeKeyPairs**
- **ec2:DescribeNatGateways**
- **ec2:DescribeNetworkAcls**
- **ec2:DescribeNetworkInterfaces**
- **ec2:DescribePrefixLists**
- **ec2:DescribeRegions**
- **ec2:DescribeRouteTables**
- **ec2:DescribeSecurityGroups**
- **ec2:DescribeSubnets**
- **ec2:DescribeTags**
- **ec2:DescribeVolumes**
- **ec2:DescribeVpcAttribute**
- **ec2:DescribeVpcClassicLink**

- **ec2:DescribeVpcClassicLinkDnsSupport**
- **ec2:DescribeVpcEndpoints**
- **ec2:DescribeVpcs**
- **ec2:GetEbsDefaultKmsKeyId**
- **ec2:ModifyInstanceAttribute**
- **ec2:ModifyNetworkInterfaceAttribute**
- **ec2:ReleaseAddress**
- **ec2:RevokeSecurityGroupEgress**
- **ec2:RevokeSecurityGroupIngress**
- **ec2:RunInstances**
- **ec2:TerminateInstances**

例 2.14. 安装过程中创建网络资源所需的权限

- **ec2:AssociateDhcpOptions**
- **ec2:AssociateRouteTable**
- **ec2:AttachInternetGateway**
- **ec2:CreateDhcpOptions**
- **ec2:CreateInternetGateway**
- **ec2:CreateNatGateway**
- **ec2:CreateRoute**
- **ec2:CreateRouteTable**
- **ec2:CreateSubnet**
- **ec2:CreateVpc**
- **ec2:CreateVpcEndpoint**
- **ec2:ModifySubnetAttribute**
- **ec2:ModifyVpcAttribute**

注意

如果您使用现有的 VPC，您的帐户不需要这些权限来创建网络资源。

例 2.15. 安装所需的 Elastic Load Balancing 权限(ELB)

- **elasticloadbalancing:AddTags**
- **elasticloadbalancing:ApplySecurityGroupsToLoadBalancer**
- **elasticloadbalancing:AttachLoadBalancerToSubnets**
- **elasticloadbalancing:ConfigureHealthCheck**
- **elasticloadbalancing:CreateLoadBalancer**
- **elasticloadbalancing:CreateLoadBalancerListeners**
- **elasticloadbalancing:DeleteLoadBalancer**
- **elasticloadbalancing:DeregisterInstancesFromLoadBalancer**
- **elasticloadbalancing:DescribeInstanceHealth**
- **elasticloadbalancing:DescribeLoadBalancerAttributes**
- **elasticloadbalancing:DescribeLoadBalancers**
- **elasticloadbalancing:DescribeTags**
- **elasticloadbalancing:ModifyLoadBalancerAttributes**
- **elasticloadbalancing:RegisterInstancesWithLoadBalancer**
- **elasticloadbalancing:SetLoadBalancerPoliciesOfListener**

例 2.16. 安装所需的 Elastic Load Balancing 权限(ELBv2)

- **elasticloadbalancing:AddTags**
- **elasticloadbalancing:CreateListener**
- **elasticloadbalancing:CreateLoadBalancer**
- **elasticloadbalancing:CreateTargetGroup**
- **elasticloadbalancing:DeleteLoadBalancer**
- **elasticloadbalancing:DeregisterTargets**
- **elasticloadbalancing:DescribeListeners**
- **elasticloadbalancing:DescribeLoadBalancerAttributes**
- **elasticloadbalancing:DescribeLoadBalancers**
- **elasticloadbalancing:DescribeTargetGroupAttributes**
- **elasticloadbalancing:DescribeTargetHealth**

- **elasticloadbalancing:ModifyLoadBalancerAttributes**
- **elasticloadbalancing:ModifyTargetGroup**
- **elasticloadbalancing:ModifyTargetGroupAttributes**
- **elasticloadbalancing:RegisterTargets**

例 2.17. 安装所需的 IAM 权限

- **iam:AddRoleToInstanceProfile**
- **iam:CreateInstanceProfile**
- **iam:CreateRole**
- **iam:DeleteInstanceProfile**
- **iam:DeleteRole**
- **iam:DeleteRolePolicy**
- **iam:GetInstanceProfile**
- **iam:GetRole**
- **iam:GetRolePolicy**
- **iam:GetUser**
- **iam:ListInstanceProfilesForRole**
- **iam:ListRoles**
- **iam:ListUsers**
- **iam:PassRole**
- **iam:PutRolePolicy**
- **iam:RemoveRoleFromInstanceProfile**
- **iam:SimulatePrincipalPolicy**
- **iam:TagRole**

注意

如果您还没有在 AWS 帐户中创建弹性负载均衡器（ELB），IAM 用户还需要 **iam:CreateServiceLinkedRole** 权限。

例 2.18. 安装所需的 Route 53 权限

- **route53:ChangeResourceRecordSets**

- **route53:ChangeTagsForResource**
- **route53:CreateHostedZone**
- **route53:DeleteHostedZone**
- **route53:GetChange**
- **route53:GetHostedZone**
- **route53:ListHostedZones**
- **route53:ListHostedZonesByName**
- **route53:ListResourceRecordSets**
- **route53:ListTagsForResource**
- **route53:UpdateHostedZoneComment**

例 2.19. 安装所需的 S3 权限

- **s3:CreateBucket**
- **s3:DeleteBucket**
- **s3:GetAccelerateConfiguration**
- **s3:GetBucketAcl**
- **s3:GetBucketCors**
- **s3:GetBucketLocation**
- **s3:GetBucketLogging**
- **s3:GetBucketObjectLockConfiguration**
- **s3:GetBucketReplication**
- **s3:GetBucketRequestPayment**
- **s3:GetBucketTagging**
- **s3:GetBucketVersioning**
- **s3:GetBucketWebsite**
- **s3:GetEncryptionConfiguration**
- **s3:GetLifecycleConfiguration**
- **s3:GetReplicationConfiguration**
- **s3:ListBucket**
- **s3:PutBucketAcl**

- **s3:PutBucketTagging**
- **s3:PutEncryptionConfiguration**

例 2.20. 集群 Operators 所需的 S3 权限

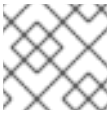
- **s3:DeleteObject**
- **s3:GetObject**
- **s3:GetObjectAcl**
- **s3:GetObjectTagging**
- **s3:GetObjectVersion**
- **s3:PutObject**
- **s3:PutObjectAcl**
- **s3:PutObjectTagging**

例 2.21. 删除基本集群资源所需的权限

- **autoscaling:DescribeAutoScalingGroups**
- **ec2:DeleteNetworkInterface**
- **ec2:DeleteVolume**
- **elasticloadbalancing:DeleteTargetGroup**
- **elasticloadbalancing:DescribeTargetGroups**
- **iam:DeleteAccessKey**
- **iam:DeleteUser**
- **iam:ListAttachedRolePolicies**
- **iam:ListInstanceProfiles**
- **iam:ListRolePolicies**
- **iam:ListUserPolicies**
- **s3:DeleteObject**
- **s3:ListBucketVersions**
- **tag:GetResources**

例 2.22. 删除网络资源所需的权限

- **ec2:DeleteDhcpOptions**
- **ec2:DeleteInternetGateway**
- **ec2:DeleteNatGateway**
- **ec2:DeleteRoute**
- **ec2:DeleteRouteTable**
- **ec2:DeleteSubnet**
- **ec2:DeleteVpc**
- **ec2:DeleteVpcEndpoints**
- **ec2:DetachInternetGateway**
- **ec2:DisassociateRouteTable**
- **ec2:ReplaceRouteTableAssociation**



注意

如果您使用现有的 VPC，您的帐户不需要这些权限来删除网络资源。

例 2.23. 创建清单所需的额外 IAM 和 S3 权限

- **iam:DeleteAccessKey**
- **iam:DeleteUser**
- **iam:DeleteUserPolicy**
- **iam:GetUserPolicy**
- **iam:ListAccessKeys**
- **iam:PutUserPolicy**
- **iam:TagUser**
- **iam:GetUserPolicy**
- **iam:ListAccessKeys**
- **s3:PutBucketPublicAccessBlock**
- **s3:GetBucketPublicAccessBlock**
- **s3:PutLifecycleConfiguration**
- **s3:HeadBucket**
- **s3:ListBucketMultipartUploads**

- **s3:AbortMultipartUpload**

注意

如果您要使用 mint 模式管理云供应商凭证，IAM 用户还需要 The **iam:CreateAccessKey and iam:CreateUser** 权限。

例 2.24. 安装时配额检查的可选权限

- **servicequotas:ListAWSDefaultServiceQuotas**

2.10.4. 获取安装程序

在安装 OpenShift Container Platform 之前，将安装文件下载到本地计算机上。

先决条件

- 运行 Linux 或 macOS 的计算机，本地磁盘空间为 500 MB

流程

1. 访问 OpenShift Cluster Manager 站点的 Infrastructure Provider 页面。如果您有红帽帐号，请使用自己的凭证登录。如果没有，请创建一个帐户。
2. 选择您的基础架构供应商。
3. 进入适用于您的安装类型的页面，下载您的操作系统的安装程序，并将文件放在要保存安装配置文件的目录中。。

重要

安装程序会在用来安装集群的计算机上创建若干文件。在完成集群安装后，您必须保留安装程序和安装程序所创建的文件。这两个文件都需要删除集群。

重要

删除安装程序创建的文件不会删除您的集群，即使集群在安装过程中失败也是如此。要删除集群，为特定云供应商完成 OpenShift Container Platform 卸载流程。

4. 提取安装程序。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ tar xvf openshift-install-linux.tar.gz
```

5. 从 Red Hat OpenShift Cluster Manager 下载安装 pull secret 。通过此 pull secret，您可以进行所含授权机构提供的服务的身份验证，这些服务包括为 OpenShift Container Platform 组件提供容器镜像的 Quay.io。

2.10.5. 生成 SSH 私钥并将其添加到代理中

如果要在集群上执行安装调试或灾难恢复，则必须为 **ssh-agent** 和安装程序提供 SSH 密钥。您可以使用此密钥访问公共集群中的 bootstrap 机器来排除安装问题。

注意

在生产环境中，您需要进行灾难恢复和调试。

您可以使用此密钥以 **core** 用户身份通过 SSH 连接到 master 节点。在部署集群时，此密钥会添加到 **core** 用户的 **~/.ssh/authorized_keys** 列表中。

注意

您必须使用一个本地密钥，而不要使用在特定平台上配置的密钥，如 AWS 密钥对。

流程

1. 如果还没有为计算机上免密码身份验证而配置的 SSH 密钥，请创建一个。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ ssh-keygen -t ed25519 -N '' \
  -f <path>/<file_name> 1
```

1 指定新 SSH 密钥的路径和文件名，如 **~/.ssh/id_rsa**。如果您已有密钥对，请确保您的公钥位于 **~/.ssh** 目录中。

运行此命令会在指定的位置生成不需要密码的 SSH 密钥。

注意

如果您计划在 **x86_64** 架构中安装使用 FIPS 验证的/Modules in Process 加密库的 OpenShift Container Platform 集群，不要创建使用 **ed25519** 算法的密钥。反之，创建一个使用 **rsa** 或 **ecdsa** 算法的密钥。

2. 作为后台任务启动 **ssh-agent** 进程：

```
$ eval "$(ssh-agent -s)"
```

输出示例

```
Agent pid 31874
```

注意

如果您的集群采用 FIPS 模式，则只使用 FIPS 兼容算法来生成 SSH 密钥。密钥必须是 RSA 或 ECDSA。

3. 将 SSH 私钥添加到 **ssh-agent**：

```
$ ssh-add <path>/<file_name> 1
```

输出示例

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

1 指定 SSH 私钥的路径和文件名，如 **~/.ssh/id_rsa**

后续步骤

- 在安装 OpenShift Container Platform 时，为安装程序提供 SSH 公钥。如果在您置备的基础架构上安装集群，您必须将此密钥提供给集群的机器。

2.10.6. 创建用于 AWS 的安装文件

要使用用户置备的基础架构在 Amazon Web Services (AWS) 上安装 OpenShift Container Platform，您必须生成并修改安装程序部署集群所需的文件，以便集群只创建要使用的机器。您要生成并自定义 **install-config.yaml** 文件、Kubernetes 清单和 Ignition 配置文件。您也可以选择在安装准备阶段首先设置独立的 **var** 分区。

2.10.6.1. 可选：创建独立 /var 分区

建议安装程序将 OpenShift Container Platform 的磁盘分区保留给安装程序。然而，在有些情况下您可能需要在文件系统的一部分中创建独立分区。

OpenShift Container Platform 支持添加单个分区来将存储附加到 **/var** 分区或 **/var** 的子目录。例如：

- **/var/lib/containers**：保存镜像相关的内容，随着更多镜像和容器添加到系统中，它所占用的存储会增加。
- **/var/lib/etcd**：保存您可能希望保持独立的数据，比如 etcd 存储的性能优化。
- **/var**：保存您希望独立保留的数据，用于特定目的（如审计）。

单独存储 **/var** 目录的内容可方便地根据需要对区域扩展存储，并可以在以后重新安装 OpenShift Container Platform 时保持该数据地完整。使用这个方法，您不必再次拉取所有容器，在更新系统时也无法复制大量日志文件。

因为 **/var** 在进行一个全新的 Red Hat Enterprise Linux CoreOS（RHCOS）安装前必需存在，所以这个流程会在 OpenShift Container Platform 安装过程的 **openshift-install** 准备阶段插入的机器配置来设置独立的 **/var** 分区。

重要

如果按照以下步骤在此流程中创建独立 **/var** 分区，则不需要再次创建 Kubernetes 清单和 Ignition 配置文件，如本节所述。

流程

1. 创建存放 OpenShift Container Platform 安装文件的目录：

```
$ mkdir $HOME/clusterconfig
```

2. 运行 **openshift-install** 在 **manifest** 和 **openshift** 子目录中创建一组文件。在出现提示时回答系统问题：

```
$ openshift-install create manifests --dir $HOME/clusterconfig
```

输出示例

```
? SSH Public Key ...
INFO Credentials loaded from the "myprofile" profile in file "/home/myuser/.aws/credentials"
INFO Consuming Install Config from target directory
INFO Manifests created in: $HOME/clusterconfig/manifests and
$HOME/clusterconfig/openshift
```

3. 可选：确认安装程序在 **clusterconfig/openshift** 目录中创建了清单：

```
$ ls $HOME/clusterconfig/openshift/
```

输出示例

```
99_kubeadmin-password-secret.yaml
99_openshift-cluster-api_master-machines-0.yaml
99_openshift-cluster-api_master-machines-1.yaml
99_openshift-cluster-api_master-machines-2.yaml
...
```

4. 创建 **MachineConfig** 对象并将其添加到 **openshift** 目录中的一个文件中。例如，把文件命名为 **98-var-partition.yaml**，将磁盘设备名称改为 **worker** 系统中存储设备的名称，并根据情况设置存储大小。这个示例将 **/var** 目录放在一个单独的分区中：

```
apiVersion: machineconfiguration.openshift.io/v1
kind: MachineConfig
metadata:
  labels:
    machineconfiguration.openshift.io/role: worker
  name: 98-var-partition
spec:
  config:
    ignition:
      version: 3.1.0
    storage:
      disks:
      - device: /dev/<device_name> 1
        partitions:
        - label: var
          startMiB: <partition_start_offset> 2
          sizeMiB: <partition_size> 3
      filesystems:
        - device: /dev/disk/by-partlabel/var
          path: /var
          format: xfs
    systemd:
      units:
        - name: var.mount 4
          enabled: true
          contents: |
            [Unit]
```

```
Before=local-fs.target
[Mount]
What=/dev/disk/by-partlabel/var
Where=/var
Options=defaults,prjquota 5
[Install]
WantedBy=local-fs.target
```

1. 要分区的磁盘的存储设备名称。
2. 当在引导磁盘中添加数据分区时，推荐最少使用 25000 MiB（Mebibytes）。root 文件系统会自动重新定义大小使其占据所有可用空间（最多到指定的偏移值）。如果没有指定值，或者指定的值小于推荐的最小值，则生成的 root 文件系统会太小，而在以后进行的 RHCOS 重新安装可能会覆盖数据分区的开始部分。
3. 数据分区的大小（以兆字节为单位）。
4. 挂载单元的名称必须与 **Where=** 指令中指定的目录匹配。例如，对于挂载在 **/var/lib/containers** 上的文件系统，该单元必须命名为 **var-lib-containers.mount**。
5. 对于用于容器存储的文件系统，必须启用 **prjquota** 挂载选项。

注意

在创建独立 **/var** 分区时，如果不同的实例类型没有相同的设备名称，则无法将不同的实例类型用于 worker 节点。

5. 再次运行 **openshift-install**，从 **manifest** 和 **openshift** 子目录中的一组文件创建 Ignition 配置：

```
$ openshift-install create ignition-configs --dir $HOME/clusterconfig
$ ls $HOME/clusterconfig/
auth  bootstrap.ign  master.ign  metadata.json  worker.ign
```

现在，您可以使用 Ignition 配置文件作为安装程序的输入来安装 Red Hat Enterprise Linux CoreOS（RHCOS）系统。

2.10.6.2. 创建安装配置文件

生成并自定义安装程序部署集群所需的安装配置文件。

先决条件

- 已获取 OpenShift Container Platform 安装程序用于用户置备的基础架构和集群的 pull secret。
- 使用红帽发布的附带 Red Hat Enterprise Linux CoreOS（RHCOS）AMI 检查您是否将集群部署到一个区域。如果您要部署到需要自定义 AMI 的区域，如 AWS GovCloud 区域，您必须手动创建 **install-config.yaml** 文件。

流程

1. 创建 **install-config.yaml** 文件。
   a. 更改到包含安装程序的目录，再运行以下命令：
      ■

```
$ ./openshift-install create install-config --dir <installation_directory> 1
```

1 对于 **<installation_directory>**，请指定用于保存安装程序所创建的文件的目录名称。

重要

指定一个空目录。一些安装信息，如 bootstrap X.509 证书，有较短的过期间隔，因此不要重复使用安装目录。如果要重复使用另一个集群安装中的个别文件，可以将其复制到您的目录中。但是，一些安装数据的文件名可能会在发行版本之间有所改变。从 OpenShift Container Platform 老版本中复制安装文件时要格外小心。

b. 在提示符处，提供您的云的配置详情：

i. 可选：选择用来访问集群机器的 SSH 密钥。

注意

对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 **ssh-agent** 进程使用的 SSH 密钥。

ii. 选择 **aws** 作为目标平台。

iii. 如果计算机上没有保存 AWS 配置集，请为您配置用于运行安装程序的用户输入 AWS 访问密钥 ID 和 secret 访问密钥。

注意

AWS 访问密钥 ID 和 secret 访问密钥存储在安装主机上当前用户主目录中的 **~/.aws/credentials** 中。如果文件中不存在导出的配置集凭证，安装程序会提示您输入凭证。您向安装程序提供的所有凭证都存储在文件中。

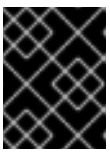
iv. 选择要将集群部署到的 AWS 区域。

v. 选择您为集群配置的 Route 53 服务的基域。

vi. 为集群输入一个描述性名称。

vii. 粘贴 Red Hat OpenShift Cluster Manager 中的 pull secret 。

2. 可选：备份 **install-config.yaml** 文件。



重要

install-config.yaml 文件会在安装过程中消耗掉。如果要重复使用此文件，必须现在备份。

其他资源

- 如需有关 AWS 配置集和凭证配置的更多信息，请参阅 AWS 文档中的配置和凭证文件设置。

2.10.6.3. 在安装过程中配置集群范围代理

生产环境可能会拒绝直接访问互联网，而是提供 HTTP 或 HTTPS 代理。您可以通过在 **install-config.yaml** 文件中配置代理设置，将新的 OpenShift Container Platform 集群配置为使用代理。

先决条件

- 您有一个现有的 **install-config.yaml** 文件。
- 您检查了集群需要访问的站点，并决定是否需要绕过代理。默认情况下代理所有集群出口流量，包括对托管云供应商 API 的调用。您需要将站点添加到 **Proxy** 对象的 **spec.noProxy** 字段来绕过代理。

注意

Proxy 对象 **status.noProxy** 字段使用安装配置中的 **networking.machineNetwork[].cidr**、**networking.clusterNetwork[].cidr** 和 **networking.serviceNetwork[]** 字段的值填充。

对于在 Amazon Web Services(AWS)、Google Cloud Platform(GCP)、Microsoft Azure 和 Red Hat OpenStack Platform(RHOSP)上安装，**Proxy** 对象 **status.noProxy** 字段也会使用实例元数据端点填充(**169.254.169.254**)。

- 如果您的集群位于 AWS 上，请将 **ec2.<region>.amazonaws.com**、**elasticloadbalancing.<region>.amazonaws.com** 和 **s3.<region>.amazonaws.com** 端点添加到 VPC 端点。需要这些端点才能完成节点到 AWS EC2 API 的请求。由于代理在容器级别而不是节点级别工作，因此您必须通过 AWS 专用网络将这些请求路由到 AWS EC2 API。在代理服务器中的允许列表中添加 EC2 API 的公共 IP 地址是不够的。

流程

1. 编辑 **install-config.yaml** 文件并添加代理设置。例如：

```
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: example.com 3
additionalTrustBundle: | 4
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
...
```

1. 用于创建集群外 HTTP 连接的代理 URL。URL 必须是 **http**。
2. 用于创建集群外 HTTPS 连接的代理 URL。
3. 要排除在代理中的目标域名、IP 地址或其他网络 CIDR 的逗号分隔列表。在域前面加 **.** 来仅匹配子域。例如：**.y.com** 匹配 **x.y.com**，但不匹配 **y.com**。使用 ***** 绕过所有目的地的代理。
4. 如果提供，安装程序会在 **openshift-config 命名空间中生成名为 user-ca- bundle** 的配置映射来保存额外的 CA 证书。如果您提供 **additionalTrustBundle** 和至少一个代理设置，则 **Proxy** 对象会被配置为引用 **trustedCA** 字段中的 **user-ca-bundle** 配置映射。然

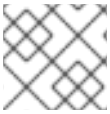
后，Cluster Network Operator 会创建一个 **trusted-ca-bundle** 配置映射，该配置映射将为 **trustedCA** 参数指定的内容与 RHCOS 信任捆绑包合并。**additionalTrustBundle** 字段是必需的，除非代理的身份证书由来自 RHCOS 信任捆绑包的颁发机构签名。

注意

安装程序不支持代理的 **readinessEndpoints** 字段。

2. 保存该文件，并在安装 OpenShift Container Platform 时引用。

安装程序会创建一个名为 **cluster** 的集群范围代理，该代理使用提供的 **install-config.yaml** 文件中的代理设置。如果没有提供代理设置，仍然会创建一个 **cluster Proxy** 对象，但它会有一个空 **spec**。



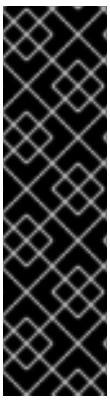
注意

只支持名为 **cluster** 的 **Proxy** 对象，且无法创建额外的代理。

2.10.6.4. 创建 Kubernetes 清单和 Ignition 配置文件

由于您必须修改一些集群定义文件并要手动启动集群机器，因此您必须生成 Kubernetes 清单和 Ignition 配置文件，集群需要这两项来创建其机器。

安装配置文件转换为 Kubernetes 清单。清单嵌套到 Ignition 配置文件中，稍后用于创建集群。



重要

- 安装程序生成的 Ignition 配置文件包含在 24 小时后过期的证书，然后在过期时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外情况是，您需要手动批准待处理的 **node-bootstrapper** 证书签名请求（CSR）来恢复 kubelet 证书。如需更多信息，请参阅*从过期的 control plane 证书中恢复*的文档。
- 建议您在生成 12 小时后使用 Ignition 配置文件，因为集群安装后 24 小时证书从 16 小时轮转至 22 小时。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中运行证书更新时避免安装失败。

先决条件

- 已获得 OpenShift Container Platform 安装程序。
- 已创建 **install-config.yaml** 安装配置文件。

流程

1. 切换到包含安装程序的目录，并为集群生成 Kubernetes 清单：

```
$ ./openshift-install create manifests --dir <installation_directory> 1
```

1 对于 **<installation_directory>**，请指定含有您创建的 **install-config.yaml** 文件的安装目录。

2. 删除定义 control plane 机器的 Kubernetes 清单文件：

```
$ rm -f <installation_directory>/openshift/99_openshift-cluster-api_master-machines-*.yaml
```

通过删除这些文件，您可以防止集群自动生成 control plane 机器。

3. 删除定义 worker 机器的 Kubernetes 清单文件：

```
$ rm -f <installation_directory>/openshift/99_openshift-cluster-api_worker-machineset-*.yaml
```

由于您要自行创建并管理 worker 机器，因此不需要初始化这些机器。

4. 检查 **<installation_directory>/manifests/cluster-scheduler-02-config.yml** Kubernetes 清单文件中的 **mastersSchedulable** 参数是否已设置为 **false**。此设置可防止在 control plane 机器上调度 pod:

   a. 打开 **<installation_directory>/manifests/cluster-scheduler-02-config.yml** 文件。

   b. 找到 **mastersSchedulable** 参数并确保它被设置为 **false**。

   c. 保存并退出文件。

5. 可选：如果您不希望 Ingress Operator 代表您创建 DNS 记录，请删除 **<installation_directory>/manifests/cluster-dns-02-config.yml** DNS 配置文件中的 **privateZone** 和 **publicZone** 部分：

```
apiVersion: config.openshift.io/v1
kind: DNS
metadata:
  creationTimestamp: null
  name: cluster
spec:
  baseDomain: example.openshift.com
  privateZone: 1
    id: mycluster-100419-private-zone
  publicZone: 2
    id: example.openshift.com
status: {}
```

1 2 完全删除此部分。

如果您这样做，后续步骤中必须手动添加入口 DNS 记录。

6. 要创建 Ignition 配置文件，从包含安装程序的目录运行以下命令：

```
$ ./openshift-install create ignition-configs --dir <installation_directory> 1
```

1 对于 **<installation_directory>**，请指定相同的安装目录。

该目录中将生成以下文件：

```
.
├── auth
│   ├── kubeadmin-password
│   └── kubeconfig
```

```
├── bootstrap.ign
├── master.ign
├── metadata.json
└── worker.ign
```

2.10.7. 提取基础架构名称

Ignition 配置文件包含一个唯一集群标识符，您可以使用它在 Amazon Web Services (AWS) 中唯一地标识您的集群。基础架构名称还用于在 OpenShift Container Platform 安装过程中定位适当的 AWS 资源。提供的 CloudFormation 模板包含对此基础架构名称的引用，因此您必须提取它。

先决条件

- 获取 OpenShift Container Platform 安装程序以及集群的 pull secret。
- 已为集群生成 Ignition 配置文件。
- 安装了 **jq** 软件包。

流程

- 要从 Ignition 配置文件元数据中提取和查看基础架构名称，请运行以下命令：

```
$ jq -r .infraID <installation_directory>/metadata.json 1
```

1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

输出示例

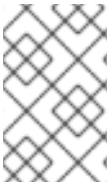
```
openshift-vw9j6 1
```

1 此命令的输出是您的集群名称和随机字符串。

2.10.8. 在 AWS 中创建 VPC

您必须在 Amazon Web Services（AWS）中创建 Virtual Private Cloud（VPC），供您的 OpenShift Container Platform 集群使用。您可以自定义 VPC 来满足您的要求，包括 VPN 和路由表。

您可以使用提供的 CloudFormation 模板和自定义参数文件创建代表 VPC 的 AWS 资源堆栈。



注意

如果不使用提供的 CloudFormation 模板来创建 AWS 基础架构，您必须检查提供的信息并手动创建基础架构。如果集群没有正确初始化，您可能需要联系红帽支持并提供您的安装日志。

先决条件

- 已配置了一个 AWS 帐户。
- 您可以通过运行 **aws configure**，将 AWS 密钥和区域添加到本地 AWS 配置集中。

- 已为集群生成 Ignition 配置文件。

流程

1. 创建一个 JSON 文件，其包含模板所需的参数值：

```
[
  {
    "ParameterKey": "VpcCidr", 1
    "ParameterValue": "10.0.0.0/16" 2
  },
  {
    "ParameterKey": "AvailabilityZoneCount", 3
    "ParameterValue": "1" 4
  },
  {
    "ParameterKey": "SubnetBits", 5
    "ParameterValue": "12" 6
  }
]
```

1. VPC 的 CIDR 块。
2. 以 **x.x.x.x/16-24** 格式指定 CIDR 块。
3. 在其中部署 VPC 的可用区的数量。
4. 指定一个 **1** 到 **3** 之间的整数。
5. 各个可用区中每个子网的大小。
6. 指定 **5** 到 **13** 之间的整数，其中 **5** 为 **/27**，**13** 为 **/19**。

2. 复制本主题的 **VPC 的 CloudFormation 模板**部分中的模板，并将它以 YAML 文件形式保存到计算机上。此模板描述了集群所需的 VPC。

3. 启动 CloudFormation 模板，以创建代表 VPC 的 AWS 资源堆栈：

重要

您必须在一行内输入命令。

```
$ aws cloudformation create-stack --stack-name <name> 1
    --template-body file://<template>.yaml 2
    --parameters file://<parameters>.json 3
```

1. **<name>** 是 CloudFormation 堆栈的名称，如 **cluster-VPC**。如果您删除集群，则需要此堆栈的名称。
2. **<template>** 是您保存的 CloudFormation 模板 YAML 文件的相对路径和名称。
3. **<parameters>** 是 CloudFormation 参数 JSON 文件的相对路径和名称。

输出示例

```
arn:aws:cloudformation:us-east-1:269333783861:stack/cluster-vpc/dbedae40-2fd3-11eb-820e-12a48460849f
```

4. 确认模板组件已存在：

```
$ aws cloudformation describe-stacks --stack-name <name>
```

在 **StackStatus** 显示 **CREATE_COMPLETE** 后，输出会显示以下参数的值。您必须将这些参数值提供给您在创建集群时要运行的其他 CloudFormation 模板：

VpcId	您的 VPC ID。
PublicSubnetIds	新公共子网的 ID。
PrivateSubnetIds	新专用子网的 ID。

2.10.8.1. VPC 的 CloudFormation 模板

您可以使用以下 CloudFormation 模板来部署 OpenShift Container Platform 集群所需的 VPC。

例 2.25. VPC 的 CloudFormation 模板

```
AWSTemplateFormatVersion: 2010-09-09
Description: Template for Best Practice VPC with 1-3 AZs

Parameters:
  VpcCidr:
    AllowedPattern: ^(([0-9]|[1-9][0-9]|1[0-9]{2}|2[0-4][0-9]|25[0-5])\.){3}([0-9]|[1-9][0-9]|1[0-9]{2}|2[0-4][0-9]|25[0-5])(\/(1[6-9]|2[0-4]))$
    ConstraintDescription: CIDR block parameter must be in the form x.x.x.x/16-24.
    Default: 10.0.0.0/16
    Description: CIDR block for VPC.
    Type: String
  AvailabilityZoneCount:
    ConstraintDescription: "The number of availability zones. (Min: 1, Max: 3)"
    MinValue: 1
    MaxValue: 3
    Default: 1
    Description: "How many AZs to create VPC subnets for. (Min: 1, Max: 3)"
    Type: Number
  SubnetBits:
    ConstraintDescription: CIDR block parameter must be in the form x.x.x.x/19-27.
    MinValue: 5
    MaxValue: 13
    Default: 12
    Description: "Size of each subnet to create within the availability zones. (Min: 5 = /27, Max: 13 = /19)"
    Type: Number
```

```
Metadata:
  AWS::CloudFormation::Interface:
    ParameterGroups:
    - Label:
        default: "Network Configuration"
      Parameters:
      - VpcCidr
      - SubnetBits
    - Label:
        default: "Availability Zones"
      Parameters:
      - AvailabilityZoneCount
    ParameterLabels:
      AvailabilityZoneCount:
        default: "Availability Zone Count"
      VpcCidr:
        default: "VPC CIDR"
      SubnetBits:
        default: "Bits Per Subnet"

Conditions:
  DoAz3: !Equals [3, !Ref AvailabilityZoneCount]
  DoAz2: !Or [!Equals [2, !Ref AvailabilityZoneCount], Condition: DoAz3]

Resources:
  VPC:
    Type: "AWS::EC2::VPC"
    Properties:
      EnableDnsSupport: "true"
      EnableDnsHostnames: "true"
      CidrBlock: !Ref VpcCidr
  PublicSubnet:
    Type: "AWS::EC2::Subnet"
    Properties:
      VpcId: !Ref VPC
      CidrBlock: !Select [0, !Cidr [!Ref VpcCidr, 6, !Ref SubnetBits]]
      AvailabilityZone: !Select
      - 0
      - Fn::GetAZs: !Ref "AWS::Region"
  PublicSubnet2:
    Type: "AWS::EC2::Subnet"
    Condition: DoAz2
    Properties:
      VpcId: !Ref VPC
      CidrBlock: !Select [1, !Cidr [!Ref VpcCidr, 6, !Ref SubnetBits]]
      AvailabilityZone: !Select
      - 1
      - Fn::GetAZs: !Ref "AWS::Region"
  PublicSubnet3:
    Type: "AWS::EC2::Subnet"
    Condition: DoAz3
    Properties:
      VpcId: !Ref VPC
      CidrBlock: !Select [2, !Cidr [!Ref VpcCidr, 6, !Ref SubnetBits]]
      AvailabilityZone: !Select
```

```
      - 2
      - Fn::GetAZs: !Ref "AWS::Region"
  InternetGateway:
    Type: "AWS::EC2::InternetGateway"
  GatewayToInternet:
    Type: "AWS::EC2::VPCGatewayAttachment"
    Properties:
      VpcId: !Ref VPC
      InternetGatewayId: !Ref InternetGateway
  PublicRouteTable:
    Type: "AWS::EC2::RouteTable"
    Properties:
      VpcId: !Ref VPC
  PublicRoute:
    Type: "AWS::EC2::Route"
    DependsOn: GatewayToInternet
    Properties:
      RouteTableId: !Ref PublicRouteTable
      DestinationCidrBlock: 0.0.0.0/0
      GatewayId: !Ref InternetGateway
  PublicSubnetRouteTableAssociation:
    Type: "AWS::EC2::SubnetRouteTableAssociation"
    Properties:
      SubnetId: !Ref PublicSubnet
      RouteTableId: !Ref PublicRouteTable
  PublicSubnetRouteTableAssociation2:
    Type: "AWS::EC2::SubnetRouteTableAssociation"
    Condition: DoAz2
    Properties:
      SubnetId: !Ref PublicSubnet2
      RouteTableId: !Ref PublicRouteTable
  PublicSubnetRouteTableAssociation3:
    Condition: DoAz3
    Type: "AWS::EC2::SubnetRouteTableAssociation"
    Properties:
      SubnetId: !Ref PublicSubnet3
      RouteTableId: !Ref PublicRouteTable
  PrivateSubnet:
    Type: "AWS::EC2::Subnet"
    Properties:
      VpcId: !Ref VPC
      CidrBlock: !Select [3, !Cidr [!Ref VpcCidr, 6, !Ref SubnetBits]]
      AvailabilityZone: !Select
      - 0
      - Fn::GetAZs: !Ref "AWS::Region"
  PrivateRouteTable:
    Type: "AWS::EC2::RouteTable"
    Properties:
      VpcId: !Ref VPC
  PrivateSubnetRouteTableAssociation:
    Type: "AWS::EC2::SubnetRouteTableAssociation"
    Properties:
      SubnetId: !Ref PrivateSubnet
      RouteTableId: !Ref PrivateRouteTable
  NAT:
    DependsOn:
```

```
  - GatewayToInternet
  Type: "AWS::EC2::NatGateway"
  Properties:
    AllocationId:
      "Fn::GetAtt":
      - EIP
      - AllocationId
    SubnetId: !Ref PublicSubnet
EIP:
  Type: "AWS::EC2::EIP"
  Properties:
    Domain: vpc
Route:
  Type: "AWS::EC2::Route"
  Properties:
    RouteTableId:
      Ref: PrivateRouteTable
    DestinationCidrBlock: 0.0.0.0/0
    NatGatewayId:
      Ref: NAT
PrivateSubnet2:
  Type: "AWS::EC2::Subnet"
  Condition: DoAz2
  Properties:
    VpcId: !Ref VPC
    CidrBlock: !Select [4, !Cidr [!Ref VpcCidr, 6, !Ref SubnetBits]]
    AvailabilityZone: !Select
    - 1
    - Fn::GetAZs: !Ref "AWS::Region"
PrivateRouteTable2:
  Type: "AWS::EC2::RouteTable"
  Condition: DoAz2
  Properties:
    VpcId: !Ref VPC
PrivateSubnetRouteTableAssociation2:
  Type: "AWS::EC2::SubnetRouteTableAssociation"
  Condition: DoAz2
  Properties:
    SubnetId: !Ref PrivateSubnet2
    RouteTableId: !Ref PrivateRouteTable2
NAT2:
  DependsOn:
  - GatewayToInternet
  Type: "AWS::EC2::NatGateway"
  Condition: DoAz2
  Properties:
    AllocationId:
      "Fn::GetAtt":
      - EIP2
      - AllocationId
    SubnetId: !Ref PublicSubnet2
EIP2:
  Type: "AWS::EC2::EIP"
  Condition: DoAz2
  Properties:
    Domain: vpc
```

```
  Route2:
    Type: "AWS::EC2::Route"
    Condition: DoAz2
    Properties:
      RouteTableId:
        Ref: PrivateRouteTable2
      DestinationCidrBlock: 0.0.0.0/0
      NatGatewayId:
        Ref: NAT2
  PrivateSubnet3:
    Type: "AWS::EC2::Subnet"
    Condition: DoAz3
    Properties:
      VpcId: !Ref VPC
      CidrBlock: !Select [5, !Cidr [!Ref VpcCidr, 6, !Ref SubnetBits]]
      AvailabilityZone: !Select
      - 2
      - Fn::GetAZs: !Ref "AWS::Region"
  PrivateRouteTable3:
    Type: "AWS::EC2::RouteTable"
    Condition: DoAz3
    Properties:
      VpcId: !Ref VPC
  PrivateSubnetRouteTableAssociation3:
    Type: "AWS::EC2::SubnetRouteTableAssociation"
    Condition: DoAz3
    Properties:
      SubnetId: !Ref PrivateSubnet3
      RouteTableId: !Ref PrivateRouteTable3
  NAT3:
    DependsOn:
    - GatewayToInternet
    Type: "AWS::EC2::NatGateway"
    Condition: DoAz3
    Properties:
      AllocationId:
        "Fn::GetAtt":
        - EIP3
        - AllocationId
      SubnetId: !Ref PublicSubnet3
  EIP3:
    Type: "AWS::EC2::EIP"
    Condition: DoAz3
    Properties:
      Domain: vpc
  Route3:
    Type: "AWS::EC2::Route"
    Condition: DoAz3
    Properties:
      RouteTableId:
        Ref: PrivateRouteTable3
      DestinationCidrBlock: 0.0.0.0/0
      NatGatewayId:
        Ref: NAT3
  S3Endpoint:
    Type: AWS::EC2::VPCEndpoint
```

```
    Properties:
      PolicyDocument:
        Version: 2012-10-17
        Statement:
        - Effect: Allow
          Principal: '*'
          Action:
          - '*'
          Resource:
          - '*'
      RouteTableIds:
      - !Ref PublicRouteTable
      - !Ref PrivateRouteTable
      - !If [DoAz2, !Ref PrivateRouteTable2, !Ref "AWS::NoValue"]
      - !If [DoAz3, !Ref PrivateRouteTable3, !Ref "AWS::NoValue"]
      ServiceName: !Join
      - ''
      - - com.amazonaws.
        - !Ref 'AWS::Region'
        - .s3
      VpcId: !Ref VPC

Outputs:
  VpcId:
    Description: ID of the new VPC.
    Value: !Ref VPC
  PublicSubnetIds:
    Description: Subnet IDs of the public subnets.
    Value:
      !Join [
        ",",
        [!Ref PublicSubnet, !If [DoAz2, !Ref PublicSubnet2, !Ref "AWS::NoValue"], !If [DoAz3, !Ref
PublicSubnet3, !Ref "AWS::NoValue"]]
      ]
  PrivateSubnetIds:
    Description: Subnet IDs of the private subnets.
    Value:
      !Join [
        ",",
        [!Ref PrivateSubnet, !If [DoAz2, !Ref PrivateSubnet2, !Ref "AWS::NoValue"], !If [DoAz3, !Ref
PrivateSubnet3, !Ref "AWS::NoValue"]]
      ]
```

其他资源

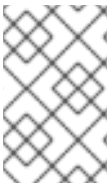
- 您可以通过导航 AWS CloudFormation 控制台来查看您创建的 CloudFormation 堆栈的详情。

2.10.9. 在 AWS 中创建网络和负载均衡组件

您必须在 OpenShift Container Platform 集群可以使用的 Amazon Web Services（AWS）中配置网络、经典或网络负载均衡。

您可以使用提供的 CloudFormation 模板和自定义参数文件来创建 AWS 资源堆栈。堆栈代表 OpenShift Container Platform 集群所需的网络和负载均衡组件。该模板还创建一个托管区和子网标签。

您可以在单一虚拟私有云(VPC)内多次运行该模板。



注意

如果不使用提供的 CloudFormation 模板来创建 AWS 基础架构，您必须检查提供的信息并手动创建基础架构。如果集群没有正确初始化，您可能需要联系红帽支持并提供您的安装日志。

先决条件

- 已配置了一个 AWS 帐户。
- 您可以通过运行 **aws configure**，将 AWS 密钥和区域添加到本地 AWS 配置集中。
- 已为集群生成 Ignition 配置文件。
- 您在 AWS 中创建并配置了 VPC 及相关子网。

流程

1. 获取您在 **install-config.yaml** 文件中为集群指定的 Route 53 基域的托管区 ID。您可以运行以下命令来获取托管区的详细信息：

```
$ aws route53 list-hosted-zones-by-name --dns-name <route53_domain> 1
```

1 对于 **<route53_domain>**，请指定您为集群生成 **install-config.yaml** 文件时所用的 Route53 基域。

输出示例

```
mycluster.example.com. False 100
HOSTEDZONES 65F8F38E-2268-B835-E15C-AB55336FCBFA
/hostedzone/Z21IXYZABCZ2A4 mycluster.example.com. 10
```

在示例输出中，托管区 ID 为 **Z21IXYZABCZ2A4**。

2. 创建一个 JSON 文件，其包含模板所需的参数值：

```
[
  {
    "ParameterKey": "ClusterName", 1
    "ParameterValue": "mycluster" 2
  },
  {
    "ParameterKey": "InfrastructureName", 3
    "ParameterValue": "mycluster-<random_string>" 4
  },
  {
    "ParameterKey": "HostedZoneId", 5
    "ParameterValue": "<random_string>" 6
  },
  {
    "ParameterKey": "HostedZoneName", 7
```

```
    "ParameterValue": "example.com" 8
  },
  {
    "ParameterKey": "PublicSubnets", 9
    "ParameterValue": "subnet-<random_string>" 10
  },
  {
    "ParameterKey": "PrivateSubnets", 11
    "ParameterValue": "subnet-<random_string>" 12
  },
  {
    "ParameterKey": "VpcId", 13
    "ParameterValue": "vpc-<random_string>" 14
  }
]
```

1. 一个简短的、代表集群的名称用于主机名等。
2. 指定您为集群生成 **install-config.yaml** 文件时所用的集群名称。
3. 您的 Ignition 配置文件中为集群编码的集群基础架构名称。
4. 指定从 Ignition 配置文件元数据中提取的基础架构名称，其格式为 **<cluster-name>-<random-string>**。
5. 用来注册目标的 Route 53 公共区 ID。
6. 指定 Route 53 公共区 ID，其格式与 **Z21IXYZABCZ2A4** 类似。您可以从 AWS 控制台获取这个值。
7. 用来注册目标的 Route 53 区。
8. 指定您为集群生成 **install-config.yaml** 文件时所用的 Route 53 基域。请勿包含 AWS 控制台中显示的结尾句点 (.)。
9. 为 VPC 创建的公共子网。
10. 指定 VPC 的 CloudFormation 模板输出的 **PublicSubnetIds** 值。
11. 为 VPC 创建的专用子网。
12. 指定 VPC 的 CloudFormation 模板输出的 **PrivateSubnetIds** 值。
13. 为集群创建的 VPC。
14. 指定 VPC 的 CloudFormation 模板输出的 **VpcId** 值。

3. 复制本主题的**网络和负载均衡器的 CloudFormation 模板**部分中的模板，并将它以 YAML 文件形式保存到计算机上。此模板描述了集群所需的网络和负载均衡对象。

重要

如果要将集群部署到 AWS 政府区域，您必须更新 CloudFormation 模板中的 **InternalApiServerRecord**，以使用 **CNAME** 记录。AWS 政府区不支持 **ALIAS** 类型的记录。

4. 启动 CloudFormation 模板，以创建 AWS 资源堆栈，该堆栈提供网络和负载均衡组件：

重要

您必须在一行内输入命令。

```
$ aws cloudformation create-stack --stack-name <name> 1
    --template-body file://<template>.yaml 2
    --parameters file://<parameters>.json 3
    --capabilities CAPABILITY_NAMED_IAM 4
```

1. **<name>** 是 CloudFormation 堆栈的名称，如 **cluster-dns**。如果您删除集群，则需要此堆栈的名称。
2. **<template>** 是您保存的 CloudFormation 模板 YAML 文件的相对路径和名称。
3. **<parameters>** 是 CloudFormation 参数 JSON 文件的相对路径和名称。
4. 您必须明确声明 **CAPABILITY_NAMED_IAM** 功能，因为提供的模板会创建一些 **AWS::IAM::Role** 资源。

输出示例

```
arn:aws:cloudformation:us-east-1:269333783861:stack/cluster-dns/cd3e5de0-2fd4-11eb-5cf0-12be5c33a183
```

5. 确认模板组件已存在：

```
$ aws cloudformation describe-stacks --stack-name <name>
```

在 **StackStatus** 显示 **CREATE_COMPLETE** 后，输出会显示以下参数的值。您必须将这些参数值提供给您在创建集群时要运行的其他 CloudFormation 模板：

PrivateHostedZoneId	专用 DNS 的托管区 ID。
ExternalApiLoadBalancerName	外部 API 负载均衡器的完整名称。
InternalApiLoadBalancerName	内部 API 负载均衡器的完整名称。
ApiServerDnsName	API 服务器的完整主机名。

RegisterNlbIpTargetsLambda	有助于为这些负载均衡器注册/撤销注册 IP 目标的 Lambda ARN。
ExternalApiTargetGroupArn	外部 API 目标组的 ARN。
InternalApiTargetGroupArn	内部 API 目标组的 ARN。
InternalServiceTargetGroupArn	内部服务目标组群的 ARN。

2.10.9.1. 网络和负载均衡器的 CloudFormation 模板

您可以使用以下 CloudFormation 模板来部署 OpenShift Container Platform 集群所需的网络对象和负载均衡器。

例 2.26. 网络和负载均衡器的 CloudFormation 模板

```
AWSTemplateFormatVersion: 2010-09-09
Description: Template for OpenShift Cluster Network Elements (Route53 & LBs)

Parameters:
  ClusterName:
    AllowedPattern: ^([a-zA-Z][a-zA-Z0-9\-]{0,26})$
    MaxLength: 27
    MinLength: 1
    ConstraintDescription: Cluster name must be alphanumeric, start with a letter, and have a
maximum of 27 characters.
    Description: A short, representative cluster name to use for host names and other identifying
names.
    Type: String
  InfrastructureName:
    AllowedPattern: ^([a-zA-Z][a-zA-Z0-9\-]{0,26})$
    MaxLength: 27
    MinLength: 1
    ConstraintDescription: Infrastructure name must be alphanumeric, start with a letter, and have a
maximum of 27 characters.
    Description: A short, unique cluster ID used to tag cloud resources and identify items owned or
used by the cluster.
    Type: String
  HostedZoneId:
    Description: The Route53 public zone ID to register the targets with, such as
Z21IXYZABCZ2A4.
    Type: String
  HostedZoneName:
```

```
    Description: The Route53 zone to register the targets with, such as example.com. Omit the
trailing period.
    Type: String
    Default: "example.com"
  PublicSubnets:
    Description: The internet-facing subnets.
    Type: List<AWS::EC2::Subnet::Id>
  PrivateSubnets:
    Description: The internal subnets.
    Type: List<AWS::EC2::Subnet::Id>
  VpcId:
    Description: The VPC-scoped resources will belong to this VPC.
    Type: AWS::EC2::VPC::Id

Metadata:
  AWS::CloudFormation::Interface:
    ParameterGroups:
    - Label:
        default: "Cluster Information"
      Parameters:
      - ClusterName
      - InfrastructureName
    - Label:
        default: "Network Configuration"
      Parameters:
      - VpcId
      - PublicSubnets
      - PrivateSubnets
    - Label:
        default: "DNS"
      Parameters:
      - HostedZoneName
      - HostedZoneId
    ParameterLabels:
      ClusterName:
        default: "Cluster Name"
      InfrastructureName:
        default: "Infrastructure Name"
      VpcId:
        default: "VPC ID"
      PublicSubnets:
        default: "Public Subnets"
      PrivateSubnets:
        default: "Private Subnets"
      HostedZoneName:
        default: "Public Hosted Zone Name"
      HostedZoneId:
        default: "Public Hosted Zone ID"

Resources:
  ExtApiElb:
    Type: AWS::ElasticLoadBalancingV2::LoadBalancer
    Properties:
      Name: !Join ["-", [!Ref InfrastructureName, "ext"]]
      IpAddressType: ipv4
      Subnets: !Ref PublicSubnets
```

```
    Type: network

IntApiElb:
  Type: AWS::ElasticLoadBalancingV2::LoadBalancer
  Properties:
    Name: !Join ["-", [!Ref InfrastructureName, "int"]]
    Scheme: internal
    IpAddressType: ipv4
    Subnets: !Ref PrivateSubnets
    Type: network

IntDns:
  Type: "AWS::Route53::HostedZone"
  Properties:
    HostedZoneConfig:
      Comment: "Managed by CloudFormation"
    Name: !Join [".", [!Ref ClusterName, !Ref HostedZoneName]]
    HostedZoneTags:
    - Key: Name
      Value: !Join ["-", [!Ref InfrastructureName, "int"]]
    - Key: !Join ["", ["kubernetes.io/cluster/", !Ref InfrastructureName]]
      Value: "owned"
    VPCs:
    - VPCId: !Ref VpcId
      VPCRegion: !Ref "AWS::Region"

ExternalApiServerRecord:
  Type: AWS::Route53::RecordSetGroup
  Properties:
    Comment: Alias record for the API server
    HostedZoneId: !Ref HostedZoneId
    RecordSets:
    - Name:
        !Join [
          ".",
          ["api", !Ref ClusterName, !Join ["", [!Ref HostedZoneName, "."]]],
        ]
      Type: A
      AliasTarget:
        HostedZoneId: !GetAtt ExtApiElb.CanonicalHostedZoneID
        DNSName: !GetAtt ExtApiElb.DNSName

InternalApiServerRecord:
  Type: AWS::Route53::RecordSetGroup
  Properties:
    Comment: Alias record for the API server
    HostedZoneId: !Ref IntDns
    RecordSets:
    - Name:
        !Join [
          ".",
          ["api", !Ref ClusterName, !Join ["", [!Ref HostedZoneName, "."]]],
        ]
      Type: A
      AliasTarget:
        HostedZoneId: !GetAtt IntApiElb.CanonicalHostedZoneID
```

```
        DNSName: !GetAtt IntApiElb.DNSName
    - Name:
        !Join [
          ".",
          ["api-int", !Ref ClusterName, !Join ["", [!Ref HostedZoneName, "."]]],
        ]
      Type: A
      AliasTarget:
        HostedZoneId: !GetAtt IntApiElb.CanonicalHostedZoneID
        DNSName: !GetAtt IntApiElb.DNSName

ExternalApiListener:
  Type: AWS::ElasticLoadBalancingV2::Listener
  Properties:
    DefaultActions:
    - Type: forward
      TargetGroupArn:
        Ref: ExternalApiTargetGroup
    LoadBalancerArn:
      Ref: ExtApiElb
    Port: 6443
    Protocol: TCP

ExternalApiTargetGroup:
  Type: AWS::ElasticLoadBalancingV2::TargetGroup
  Properties:
    HealthCheckIntervalSeconds: 10
    HealthCheckPath: "/readyz"
    HealthCheckPort: 6443
    HealthCheckProtocol: HTTPS
    HealthyThresholdCount: 2
    UnhealthyThresholdCount: 2
    Port: 6443
    Protocol: TCP
    TargetType: ip
    VpcId:
      Ref: VpcId
    TargetGroupAttributes:
    - Key: deregistration_delay.timeout_seconds
      Value: 60

InternalApiListener:
  Type: AWS::ElasticLoadBalancingV2::Listener
  Properties:
    DefaultActions:
    - Type: forward
      TargetGroupArn:
        Ref: InternalApiTargetGroup
    LoadBalancerArn:
      Ref: IntApiElb
    Port: 6443
    Protocol: TCP

InternalApiTargetGroup:
  Type: AWS::ElasticLoadBalancingV2::TargetGroup
  Properties:
```

```
    HealthCheckIntervalSeconds: 10
    HealthCheckPath: "/readyz"
    HealthCheckPort: 6443
    HealthCheckProtocol: HTTPS
    HealthyThresholdCount: 2
    UnhealthyThresholdCount: 2
    Port: 6443
    Protocol: TCP
    TargetType: ip
    VpcId:
      Ref: VpcId
    TargetGroupAttributes:
    - Key: deregistration_delay.timeout_seconds
      Value: 60

InternalServiceInternalListener:
  Type: AWS::ElasticLoadBalancingV2::Listener
  Properties:
    DefaultActions:
    - Type: forward
      TargetGroupArn:
        Ref: InternalServiceTargetGroup
    LoadBalancerArn:
      Ref: IntApiElb
    Port: 22623
    Protocol: TCP

InternalServiceTargetGroup:
  Type: AWS::ElasticLoadBalancingV2::TargetGroup
  Properties:
    HealthCheckIntervalSeconds: 10
    HealthCheckPath: "/healthz"
    HealthCheckPort: 22623
    HealthCheckProtocol: HTTPS
    HealthyThresholdCount: 2
    UnhealthyThresholdCount: 2
    Port: 22623
    Protocol: TCP
    TargetType: ip
    VpcId:
      Ref: VpcId
    TargetGroupAttributes:
    - Key: deregistration_delay.timeout_seconds
      Value: 60

RegisterTargetLambdaIamRole:
  Type: AWS::IAM::Role
  Properties:
    RoleName: !Join ["-", [!Ref InfrastructureName, "nlb", "lambda", "role"]]
    AssumeRolePolicyDocument:
      Version: "2012-10-17"
      Statement:
      - Effect: "Allow"
        Principal:
          Service:
          - "lambda.amazonaws.com"
```

```
          Action:
          - "sts:AssumeRole"
      Path: "/"
      Policies:
      - PolicyName: !Join ["-", [!Ref InfrastructureName, "master", "policy"]]
        PolicyDocument:
          Version: "2012-10-17"
          Statement:
          - Effect: "Allow"
            Action:
              [
                "elasticloadbalancing:RegisterTargets",
                "elasticloadbalancing:DeregisterTargets",
              ]
            Resource: !Ref InternalApiTargetGroup
          - Effect: "Allow"
            Action:
              [
                "elasticloadbalancing:RegisterTargets",
                "elasticloadbalancing:DeregisterTargets",
              ]
            Resource: !Ref InternalServiceTargetGroup
          - Effect: "Allow"
            Action:
              [
                "elasticloadbalancing:RegisterTargets",
                "elasticloadbalancing:DeregisterTargets",
              ]
            Resource: !Ref ExternalApiTargetGroup

    RegisterNlbIpTargets:
      Type: "AWS::Lambda::Function"
      Properties:
        Handler: "index.handler"
        Role:
          Fn::GetAtt:
          - "RegisterTargetLambdaIamRole"
          - "Arn"
        Code:
          ZipFile: |
            import json
            import boto3
            import cfnresponse
            def handler(event, context):
              elb = boto3.client('elbv2')
              if event['RequestType'] == 'Delete':
                elb.deregister_targets(TargetGroupArn=event['ResourceProperties']
['TargetArn'],Targets=[{'Id': event['ResourceProperties']['TargetIp']}])
              elif event['RequestType'] == 'Create':
                elb.register_targets(TargetGroupArn=event['ResourceProperties']['TargetArn'],Targets=
[{'Id': event['ResourceProperties']['TargetIp']}])
              responseData = {}
              cfnresponse.send(event, context, cfnresponse.SUCCESS, responseData,
event['ResourceProperties']['TargetArn']+event['ResourceProperties']['TargetIp'])
        Runtime: "python3.7"
        Timeout: 120
```

```
  RegisterSubnetTagsLambdaIamRole:
    Type: AWS::IAM::Role
    Properties:
      RoleName: !Join ["-", [!Ref InfrastructureName, "subnet-tags-lambda-role"]]
      AssumeRolePolicyDocument:
        Version: "2012-10-17"
        Statement:
        - Effect: "Allow"
          Principal:
            Service:
            - "lambda.amazonaws.com"
          Action:
          - "sts:AssumeRole"
      Path: "/"
      Policies:
      - PolicyName: !Join ["-", [!Ref InfrastructureName, "subnet-tagging-policy"]]
        PolicyDocument:
          Version: "2012-10-17"
          Statement:
          - Effect: "Allow"
            Action:
              [
                "ec2:DeleteTags",
                "ec2:CreateTags"
              ]
            Resource: "arn:aws:ec2:*:*:subnet/*"
          - Effect: "Allow"
            Action:
              [
                "ec2:DescribeSubnets",
                "ec2:DescribeTags"
              ]
            Resource: "*"

  RegisterSubnetTags:
    Type: "AWS::Lambda::Function"
    Properties:
      Handler: "index.handler"
      Role:
        Fn::GetAtt:
        - "RegisterSubnetTagsLambdaIamRole"
        - "Arn"
      Code:
        ZipFile: |
          import json
          import boto3
          import cfnresponse
          def handler(event, context):
            ec2_client = boto3.client('ec2')
            if event['RequestType'] == 'Delete':
              for subnet_id in event['ResourceProperties']['Subnets']:
                ec2_client.delete_tags(Resources=[subnet_id], Tags=[{'Key': 'kubernetes.io/cluster/' +
event['ResourceProperties']['InfrastructureName']}]);
            elif event['RequestType'] == 'Create':
              for subnet_id in event['ResourceProperties']['Subnets']:
```

```
            ec2_client.create_tags(Resources=[subnet_id], Tags=[{'Key': 'kubernetes.io/cluster/' +
event['ResourceProperties']['InfrastructureName'], 'Value': 'shared'}]);
          responseData = {}
          cfnresponse.send(event, context, cfnresponse.SUCCESS, responseData,
event['ResourceProperties']['InfrastructureName']+event['ResourceProperties']['Subnets'][0])
      Runtime: "python3.7"
      Timeout: 120

  RegisterPublicSubnetTags:
    Type: Custom::SubnetRegister
    Properties:
      ServiceToken: !GetAtt RegisterSubnetTags.Arn
      InfrastructureName: !Ref InfrastructureName
      Subnets: !Ref PublicSubnets

  RegisterPrivateSubnetTags:
    Type: Custom::SubnetRegister
    Properties:
      ServiceToken: !GetAtt RegisterSubnetTags.Arn
      InfrastructureName: !Ref InfrastructureName
      Subnets: !Ref PrivateSubnets

Outputs:
  PrivateHostedZoneId:
    Description: Hosted zone ID for the private DNS, which is required for private records.
    Value: !Ref IntDns
  ExternalApiLoadBalancerName:
    Description: Full name of the external API load balancer.
    Value: !GetAtt ExtApiElb.LoadBalancerFullName
  InternalApiLoadBalancerName:
    Description: Full name of the internal API load balancer.
    Value: !GetAtt IntApiElb.LoadBalancerFullName
  ApiServerDnsName:
    Description: Full hostname of the API server, which is required for the Ignition config files.
    Value: !Join [".", ["api-int", !Ref ClusterName, !Ref HostedZoneName]]
  RegisterNlbIpTargetsLambda:
    Description: Lambda ARN useful to help register or deregister IP targets for these load
balancers.
    Value: !GetAtt RegisterNlbIpTargets.Arn
  ExternalApiTargetGroupArn:
    Description: ARN of the external API target group.
    Value: !Ref ExternalApiTargetGroup
  InternalApiTargetGroupArn:
    Description: ARN of the internal API target group.
    Value: !Ref InternalApiTargetGroup
  InternalServiceTargetGroupArn:
    Description: ARN of the internal service target group.
    Value: !Ref InternalServiceTargetGroup
```

重要

如果要将集群部署到 AWS 政府区域，您必须更新 **InternalApiServerRecord** 以使用 **CNAME** 记录。AWS 政府区不支持 **ALIAS** 类型的记录。例如：

```
Type: CNAME
TTL: 10
ResourceRecords:
- !GetAtt IntApiElb.DNSName
```

其他资源

- 您可以通过导航 AWS CloudFormation 控制台来查看您创建的 CloudFormation 堆栈的详情。
- 您可以通过导航到 AWS Route 53 控制台 来查看托管区的详情。
- 有关列出公共托管区的更多信息，请参阅 AWS 文档中的列出公共托管区。

2.10.10. 在 AWS 中创建安全组和角色

您必须在 Amazon Web Services (AWS) 中创建安全组和角色，供您的 OpenShift Container Platform 集群使用。

您可以使用提供的 CloudFormation 模板和自定义参数文件来创建 AWS 资源堆栈。堆栈代表 OpenShift Container Platform 集群所需的安全组和角色。

注意

如果不使用提供的 CloudFormation 模板来创建 AWS 基础架构，您必须检查提供的信息并手动创建基础架构。如果集群没有正确初始化，您可能需要联系红帽支持并提供您的安装日志。

先决条件

- 已配置了一个 AWS 帐户。
- 您可以通过运行 **aws configure**，将 AWS 密钥和区域添加到本地 AWS 配置集中。
- 已为集群生成 Ignition 配置文件。
- 您在 AWS 中创建并配置了 VPC 及相关子网。

流程

1. 创建一个 JSON 文件，其包含模板所需的参数值：

```
[
  {
    "ParameterKey": "InfrastructureName", 1
    "ParameterValue": "mycluster-<random_string>" 2
  },
  {
    "ParameterKey": "VpcCidr", 3
    "ParameterValue": "10.0.0.0/16" 4
```

```
  },
  {
    "ParameterKey": "PrivateSubnets", 5
    "ParameterValue": "subnet-<random_string>" 6
  },
  {
    "ParameterKey": "VpcId", 7
    "ParameterValue": "vpc-<random_string>" 8
  }
]
```

1. 您的 Ignition 配置文件中为集群编码的集群基础架构名称。
2. 指定从 Ignition 配置文件元数据中提取的基础架构名称，其格式为 **<cluster-name>-<random-string>**。
3. VPC 的 CIDR 块。
4. 指定以 **x.x.x.x/16-24** 格式定义的用于 VPC 的 CIDR 地址块。
5. 为 VPC 创建的专用子网。
6. 指定 VPC 的 CloudFormation 模板输出的 **PrivateSubnetIds** 值。
7. 为集群创建的 VPC。
8. 指定 VPC 的 CloudFormation 模板输出的 **VpcId** 值。

2. 复制本主题的**安全对象的 CloudFormation 模板**部分中的模板，并将它以 YAML 文件形式保存到计算机上。此模板描述了集群所需的安全组和角色。
3. 启动 CloudFormation 模板，以创建代表安全组和角色的 AWS 资源堆栈：

重要

您必须在一行内输入命令。

```
$ aws cloudformation create-stack --stack-name <name> 1
    --template-body file://<template>.yaml 2
    --parameters file://<parameters>.json 3
    --capabilities CAPABILITY_NAMED_IAM 4
```

1. **<name>** 是 CloudFormation 堆栈的名称，如 **cluster-sec**。如果您删除集群，则需要此堆栈的名称。
2. **<template>** 是您保存的 CloudFormation 模板 YAML 文件的相对路径和名称。
3. **<parameters>** 是 CloudFormation 参数 JSON 文件的相对路径和名称。
4. 您必须明确声明 **CAPABILITY_NAMED_IAM** 功能，因为提供的模板会创建一些 **AWS::IAM::Role** 和 **AWS::IAM::InstanceProfile** 资源。

输出示例

```
arn:aws:cloudformation:us-east-1:269333783861:stack/cluster-sec/03bd4210-2ed7-11eb-6d7a-13fc0b61e9db
```

4. 确认模板组件已存在：

```
$ aws cloudformation describe-stacks --stack-name <name>
```

在 **StackStatus** 显示 **CREATE_COMPLETE** 后，输出会显示以下参数的值。您必须将这些参数值提供给您在创建集群时要运行的其他 CloudFormation 模板：

MasterSecurityGroupId	Master 安全组 ID
WorkerSecurityGroupId	worker 安全组 ID
MasterInstanceProfile	Master IAM 实例配置集
WorkerInstanceProfile	worker IAM 实例配置集

2.10.10.1. 安全对象的 CloudFormation 模板

您可以使用以下 CloudFormation 模板来部署 OpenShift Container Platform 集群所需的安全对象。

例 2.27. 安全对象的 CloudFormation 模板

```
AWSTemplateFormatVersion: 2010-09-09
Description: Template for OpenShift Cluster Security Elements (Security Groups & IAM)

Parameters:
  InfrastructureName:
    AllowedPattern: ^([a-zA-Z][a-zA-Z0-9\-]{0,26})$
    MaxLength: 27
    MinLength: 1
    ConstraintDescription: Infrastructure name must be alphanumeric, start with a letter, and have a maximum of 27 characters.
    Description: A short, unique cluster ID used to tag cloud resources and identify items owned or used by the cluster.
    Type: String
  VpcCidr:
    AllowedPattern: ^(([0-9]|[1-9][0-9]|1[0-9]{2}|2[0-4][0-9]|25[0-5])\.){3}([0-9]|[1-9][0-9]|1[0-9]{2}|2[0-4][0-9]|25[0-5])(\/(1[6-9]|2[0-4]))$
    ConstraintDescription: CIDR block parameter must be in the form x.x.x.x/16-24.
    Default: 10.0.0.0/16
    Description: CIDR block for VPC.
    Type: String
```

```
  VpcId:
    Description: The VPC-scoped resources will belong to this VPC.
    Type: AWS::EC2::VPC::Id
  PrivateSubnets:
    Description: The internal subnets.
    Type: List<AWS::EC2::Subnet::Id>

Metadata:
  AWS::CloudFormation::Interface:
    ParameterGroups:
    - Label:
        default: "Cluster Information"
      Parameters:
      - InfrastructureName
    - Label:
        default: "Network Configuration"
      Parameters:
      - VpcId
      - VpcCidr
      - PrivateSubnets
    ParameterLabels:
      InfrastructureName:
        default: "Infrastructure Name"
      VpcId:
        default: "VPC ID"
      VpcCidr:
        default: "VPC CIDR"
      PrivateSubnets:
        default: "Private Subnets"

Resources:
  MasterSecurityGroup:
    Type: AWS::EC2::SecurityGroup
    Properties:
      GroupDescription: Cluster Master Security Group
      SecurityGroupIngress:
      - IpProtocol: icmp
        FromPort: 0
        ToPort: 0
        CidrIp: !Ref VpcCidr
      - IpProtocol: tcp
        FromPort: 22
        ToPort: 22
        CidrIp: !Ref VpcCidr
      - IpProtocol: tcp
        ToPort: 6443
        FromPort: 6443
        CidrIp: !Ref VpcCidr
      - IpProtocol: tcp
        FromPort: 22623
        ToPort: 22623
        CidrIp: !Ref VpcCidr
      VpcId: !Ref VpcId

  WorkerSecurityGroup:
    Type: AWS::EC2::SecurityGroup
```

```
  Properties:
    GroupDescription: Cluster Worker Security Group
    SecurityGroupIngress:
    - IpProtocol: icmp
      FromPort: 0
      ToPort: 0
      CidrIp: !Ref VpcCidr
    - IpProtocol: tcp
      FromPort: 22
      ToPort: 22
      CidrIp: !Ref VpcCidr
    VpcId: !Ref VpcId

MasterIngressEtcd:
  Type: AWS::EC2::SecurityGroupIngress
  Properties:
    GroupId: !GetAtt MasterSecurityGroup.GroupId
    SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId
    Description: etcd
    FromPort: 2379
    ToPort: 2380
    IpProtocol: tcp

MasterIngressVxlan:
  Type: AWS::EC2::SecurityGroupIngress
  Properties:
    GroupId: !GetAtt MasterSecurityGroup.GroupId
    SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId
    Description: Vxlan packets
    FromPort: 4789
    ToPort: 4789
    IpProtocol: udp

MasterIngressWorkerVxlan:
  Type: AWS::EC2::SecurityGroupIngress
  Properties:
    GroupId: !GetAtt MasterSecurityGroup.GroupId
    SourceSecurityGroupId: !GetAtt WorkerSecurityGroup.GroupId
    Description: Vxlan packets
    FromPort: 4789
    ToPort: 4789
    IpProtocol: udp

MasterIngressGeneve:
  Type: AWS::EC2::SecurityGroupIngress
  Properties:
    GroupId: !GetAtt MasterSecurityGroup.GroupId
    SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId
    Description: Geneve packets
    FromPort: 6081
    ToPort: 6081
    IpProtocol: udp

MasterIngressWorkerGeneve:
  Type: AWS::EC2::SecurityGroupIngress
  Properties:
```

```
      GroupId: !GetAtt MasterSecurityGroup.GroupId
      SourceSecurityGroupId: !GetAtt WorkerSecurityGroup.GroupId
      Description: Geneve packets
      FromPort: 6081
      ToPort: 6081
      IpProtocol: udp

  MasterIngressInternal:
    Type: AWS::EC2::SecurityGroupIngress
    Properties:
      GroupId: !GetAtt MasterSecurityGroup.GroupId
      SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId
      Description: Internal cluster communication
      FromPort: 9000
      ToPort: 9999
      IpProtocol: tcp

  MasterIngressWorkerInternal:
    Type: AWS::EC2::SecurityGroupIngress
    Properties:
      GroupId: !GetAtt MasterSecurityGroup.GroupId
      SourceSecurityGroupId: !GetAtt WorkerSecurityGroup.GroupId
      Description: Internal cluster communication
      FromPort: 9000
      ToPort: 9999
      IpProtocol: tcp

  MasterIngressInternalUDP:
    Type: AWS::EC2::SecurityGroupIngress
    Properties:
      GroupId: !GetAtt MasterSecurityGroup.GroupId
      SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId
      Description: Internal cluster communication
      FromPort: 9000
      ToPort: 9999
      IpProtocol: udp

  MasterIngressWorkerInternalUDP:
    Type: AWS::EC2::SecurityGroupIngress
    Properties:
      GroupId: !GetAtt MasterSecurityGroup.GroupId
      SourceSecurityGroupId: !GetAtt WorkerSecurityGroup.GroupId
      Description: Internal cluster communication
      FromPort: 9000
      ToPort: 9999
      IpProtocol: udp

  MasterIngressKube:
    Type: AWS::EC2::SecurityGroupIngress
    Properties:
      GroupId: !GetAtt MasterSecurityGroup.GroupId
      SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId
      Description: Kubernetes kubelet, scheduler and controller manager
      FromPort: 10250
      ToPort: 10259
      IpProtocol: tcp
```

```
MasterIngressWorkerKube:
  Type: AWS::EC2::SecurityGroupIngress
  Properties:
    GroupId: !GetAtt MasterSecurityGroup.GroupId
    SourceSecurityGroupId: !GetAtt WorkerSecurityGroup.GroupId
    Description: Kubernetes kubelet, scheduler and controller manager
    FromPort: 10250
    ToPort: 10259
    IpProtocol: tcp

MasterIngressIngressServices:
  Type: AWS::EC2::SecurityGroupIngress
  Properties:
    GroupId: !GetAtt MasterSecurityGroup.GroupId
    SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId
    Description: Kubernetes ingress services
    FromPort: 30000
    ToPort: 32767
    IpProtocol: tcp

MasterIngressWorkerIngressServices:
  Type: AWS::EC2::SecurityGroupIngress
  Properties:
    GroupId: !GetAtt MasterSecurityGroup.GroupId
    SourceSecurityGroupId: !GetAtt WorkerSecurityGroup.GroupId
    Description: Kubernetes ingress services
    FromPort: 30000
    ToPort: 32767
    IpProtocol: tcp

MasterIngressIngressServicesUDP:
  Type: AWS::EC2::SecurityGroupIngress
  Properties:
    GroupId: !GetAtt MasterSecurityGroup.GroupId
    SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId
    Description: Kubernetes ingress services
    FromPort: 30000
    ToPort: 32767
    IpProtocol: udp

MasterIngressWorkerIngressServicesUDP:
  Type: AWS::EC2::SecurityGroupIngress
  Properties:
    GroupId: !GetAtt MasterSecurityGroup.GroupId
    SourceSecurityGroupId: !GetAtt WorkerSecurityGroup.GroupId
    Description: Kubernetes ingress services
    FromPort: 30000
    ToPort: 32767
    IpProtocol: udp

WorkerIngressVxlan:
  Type: AWS::EC2::SecurityGroupIngress
  Properties:
    GroupId: !GetAtt WorkerSecurityGroup.GroupId
    SourceSecurityGroupId: !GetAtt WorkerSecurityGroup.GroupId
```

```
      Description: Vxlan packets
      FromPort: 4789
      ToPort: 4789
      IpProtocol: udp

  WorkerIngressMasterVxlan:
    Type: AWS::EC2::SecurityGroupIngress
    Properties:
      GroupId: !GetAtt WorkerSecurityGroup.GroupId
      SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId
      Description: Vxlan packets
      FromPort: 4789
      ToPort: 4789
      IpProtocol: udp

  WorkerIngressGeneve:
    Type: AWS::EC2::SecurityGroupIngress
    Properties:
      GroupId: !GetAtt WorkerSecurityGroup.GroupId
      SourceSecurityGroupId: !GetAtt WorkerSecurityGroup.GroupId
      Description: Geneve packets
      FromPort: 6081
      ToPort: 6081
      IpProtocol: udp

  WorkerIngressMasterGeneve:
    Type: AWS::EC2::SecurityGroupIngress
    Properties:
      GroupId: !GetAtt WorkerSecurityGroup.GroupId
      SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId
      Description: Geneve packets
      FromPort: 6081
      ToPort: 6081
      IpProtocol: udp

  WorkerIngressInternal:
    Type: AWS::EC2::SecurityGroupIngress
    Properties:
      GroupId: !GetAtt WorkerSecurityGroup.GroupId
      SourceSecurityGroupId: !GetAtt WorkerSecurityGroup.GroupId
      Description: Internal cluster communication
      FromPort: 9000
      ToPort: 9999
      IpProtocol: tcp

  WorkerIngressMasterInternal:
    Type: AWS::EC2::SecurityGroupIngress
    Properties:
      GroupId: !GetAtt WorkerSecurityGroup.GroupId
      SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId
      Description: Internal cluster communication
      FromPort: 9000
      ToPort: 9999
      IpProtocol: tcp

  WorkerIngressInternalUDP:
```

```
  Type: AWS::EC2::SecurityGroupIngress
  Properties:
    GroupId: !GetAtt WorkerSecurityGroup.GroupId
    SourceSecurityGroupId: !GetAtt WorkerSecurityGroup.GroupId
    Description: Internal cluster communication
    FromPort: 9000
    ToPort: 9999
    IpProtocol: udp

WorkerIngressMasterInternalUDP:
  Type: AWS::EC2::SecurityGroupIngress
  Properties:
    GroupId: !GetAtt WorkerSecurityGroup.GroupId
    SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId
    Description: Internal cluster communication
    FromPort: 9000
    ToPort: 9999
    IpProtocol: udp

WorkerIngressKube:
  Type: AWS::EC2::SecurityGroupIngress
  Properties:
    GroupId: !GetAtt WorkerSecurityGroup.GroupId
    SourceSecurityGroupId: !GetAtt WorkerSecurityGroup.GroupId
    Description: Kubernetes secure kubelet port
    FromPort: 10250
    ToPort: 10250
    IpProtocol: tcp

WorkerIngressWorkerKube:
  Type: AWS::EC2::SecurityGroupIngress
  Properties:
    GroupId: !GetAtt WorkerSecurityGroup.GroupId
    SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId
    Description: Internal Kubernetes communication
    FromPort: 10250
    ToPort: 10250
    IpProtocol: tcp

WorkerIngressIngressServices:
  Type: AWS::EC2::SecurityGroupIngress
  Properties:
    GroupId: !GetAtt WorkerSecurityGroup.GroupId
    SourceSecurityGroupId: !GetAtt WorkerSecurityGroup.GroupId
    Description: Kubernetes ingress services
    FromPort: 30000
    ToPort: 32767
    IpProtocol: tcp

WorkerIngressMasterIngressServices:
  Type: AWS::EC2::SecurityGroupIngress
  Properties:
    GroupId: !GetAtt WorkerSecurityGroup.GroupId
    SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId
    Description: Kubernetes ingress services
    FromPort: 30000
```

```
      ToPort: 32767
      IpProtocol: tcp

  WorkerIngressIngressServicesUDP:
    Type: AWS::EC2::SecurityGroupIngress
    Properties:
      GroupId: !GetAtt WorkerSecurityGroup.GroupId
      SourceSecurityGroupId: !GetAtt WorkerSecurityGroup.GroupId
      Description: Kubernetes ingress services
      FromPort: 30000
      ToPort: 32767
      IpProtocol: udp

  WorkerIngressMasterIngressServicesUDP:
    Type: AWS::EC2::SecurityGroupIngress
    Properties:
      GroupId: !GetAtt WorkerSecurityGroup.GroupId
      SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId
      Description: Kubernetes ingress services
      FromPort: 30000
      ToPort: 32767
      IpProtocol: udp

  MasterIamRole:
    Type: AWS::IAM::Role
    Properties:
      AssumeRolePolicyDocument:
        Version: "2012-10-17"
        Statement:
        - Effect: "Allow"
          Principal:
            Service:
            - "ec2.amazonaws.com"
          Action:
          - "sts:AssumeRole"
      Policies:
      - PolicyName: !Join ["-", [!Ref InfrastructureName, "master", "policy"]]
        PolicyDocument:
          Version: "2012-10-17"
          Statement:
          - Effect: "Allow"
            Action:
            - "ec2:AttachVolume"
            - "ec2:AuthorizeSecurityGroupIngress"
            - "ec2:CreateSecurityGroup"
            - "ec2:CreateTags"
            - "ec2:CreateVolume"
            - "ec2:DeleteSecurityGroup"
            - "ec2:DeleteVolume"
            - "ec2:Describe*"
            - "ec2:DetachVolume"
            - "ec2:ModifyInstanceAttribute"
            - "ec2:ModifyVolume"
            - "ec2:RevokeSecurityGroupIngress"
            - "elasticloadbalancing:AddTags"
            - "elasticloadbalancing:AttachLoadBalancerToSubnets"
```

```
          - "elasticloadbalancing:ApplySecurityGroupsToLoadBalancer"
          - "elasticloadbalancing:CreateListener"
          - "elasticloadbalancing:CreateLoadBalancer"
          - "elasticloadbalancing:CreateLoadBalancerPolicy"
          - "elasticloadbalancing:CreateLoadBalancerListeners"
          - "elasticloadbalancing:CreateTargetGroup"
          - "elasticloadbalancing:ConfigureHealthCheck"
          - "elasticloadbalancing:DeleteListener"
          - "elasticloadbalancing:DeleteLoadBalancer"
          - "elasticloadbalancing:DeleteLoadBalancerListeners"
          - "elasticloadbalancing:DeleteTargetGroup"
          - "elasticloadbalancing:DeregisterInstancesFromLoadBalancer"
          - "elasticloadbalancing:DeregisterTargets"
          - "elasticloadbalancing:Describe*"
          - "elasticloadbalancing:DetachLoadBalancerFromSubnets"
          - "elasticloadbalancing:ModifyListener"
          - "elasticloadbalancing:ModifyLoadBalancerAttributes"
          - "elasticloadbalancing:ModifyTargetGroup"
          - "elasticloadbalancing:ModifyTargetGroupAttributes"
          - "elasticloadbalancing:RegisterInstancesWithLoadBalancer"
          - "elasticloadbalancing:RegisterTargets"
          - "elasticloadbalancing:SetLoadBalancerPoliciesForBackendServer"
          - "elasticloadbalancing:SetLoadBalancerPoliciesOfListener"
          - "kms:DescribeKey"
          Resource: "*"

MasterInstanceProfile:
  Type: "AWS::IAM::InstanceProfile"
  Properties:
    Roles:
    - Ref: "MasterIamRole"

WorkerIamRole:
  Type: AWS::IAM::Role
  Properties:
    AssumeRolePolicyDocument:
      Version: "2012-10-17"
      Statement:
      - Effect: "Allow"
        Principal:
          Service:
          - "ec2.amazonaws.com"
        Action:
        - "sts:AssumeRole"
    Policies:
    - PolicyName: !Join ["-", [!Ref InfrastructureName, "worker", "policy"]]
      PolicyDocument:
        Version: "2012-10-17"
        Statement:
        - Effect: "Allow"
          Action:
          - "ec2:DescribeInstances"
          - "ec2:DescribeRegions"
          Resource: "*"

WorkerInstanceProfile:
```

```
      Type: "AWS::IAM::InstanceProfile"
      Properties:
        Roles:
        - Ref: "WorkerIamRole"

  Outputs:
    MasterSecurityGroupId:
      Description: Master Security Group ID
      Value: !GetAtt MasterSecurityGroup.GroupId

    WorkerSecurityGroupId:
      Description: Worker Security Group ID
      Value: !GetAtt WorkerSecurityGroup.GroupId

    MasterInstanceProfile:
      Description: Master IAM Instance Profile
      Value: !Ref MasterInstanceProfile

    WorkerInstanceProfile:
      Description: Worker IAM Instance Profile
      Value: !Ref WorkerInstanceProfile
```

其他资源

- 您可以通过导航 AWS CloudFormation 控制台来查看您创建的 CloudFormation 堆栈的详情。

2.10.11. AWS 基础架构的 RHCOS AMI

红帽为您为 OpenShift Container Platform 节点指定的各种 Amazon Web Services（AWS）区域提供了有效的 Red Hat Enterprise Linux CoreOS（RHCOS）AMI。

注意

您还可以导入您自己的 AMI，来安装到没有发布 RHCOS AMI 的区域。

表 2.31. RHCOS AMI

AWS 区	AWS AMI
af-south-1	**ami-09921c9c1c36e695c**
ap-east-1	**ami-01ee8446e9af6b197**
ap-northeast-1	**ami-04e5b5722a55846ea**
ap-northeast-2	**ami-0fdc25c8a0273a742**
ap-south-1	**ami-09e3deb397cc526a8**
ap-southeast-1	**ami-0630e03f75e02eec4**

AWS 区	AWS AMI
ap-southeast-2	**ami-069450613262ba03c**
ca-central-1	**ami-012518cdbd3057dfd**
eu-central-1	**ami-0bd7175ff5b1aef0c**
eu-north-1	**ami-06c9ec42d0a839ad2**
eu-south-1	**ami-0614d7440a0363d71**
eu-west-1	**ami-01b89df58b5d4d5fa**
eu-west-2	**ami-06f6e31ddd554f89d**
eu-west-3	**ami-0dc82e2517ded15a1**
me-south-1	**ami-07d181e3aa0f76067**
sa-east-1	**ami-0cd44e6dd20e6c7fa**
us-east-1	**ami-04a16d506e5b0e246**
us-east-2	**ami-0a1f868ad58ea59a7**
us-west-1	**ami-0a65d76e3a6f6622f**
us-west-2	**ami-0dd9008abadc519f1**

2.10.11.1. 没有公布的 RHCOS AMI 的 AWS 区域

您可以将 OpenShift Container Platform 集群部署到 Amazon Web Services（AWS）区域，而无需对 Red Hat Enterprise Linux CoreOS（RHCOS）Amazon Machine Image（AMI）或 AWS 软件开发 kit（SDK）的原生支持。如果 AWS 区域没有可用的已公布的 AMI，您可以在安装集群前上传自定义 AMI。如果您要将集群部署到 AWS 政府区域，则需要此参数。

如果您部署到没有公布的 RHCOS AMI 的非机构区域，且您没有指定自定义的 AMI，安装程序会自动将 **us-east-1** AMI 复制到用户帐户。然后，安装程序使用默认或用户指定的密钥管理服务（KMS）密钥创建带有加密 EBS 卷的 control plane 机器。这允许 AMI 跟踪与公布的 RHCOS AMI 相同的进程工作流。

在集群创建过程中，无法从终端中选择没有原生支持 RHCOS AMI 的区域，因为它没有发布。但是，您可以通过在 **install-config.yaml** 文件中配置自定义 AMI 来安装到这个区域。

2.10.11.2. 在 AWS 中上传自定义 RHCOS AMI

如果要部署到自定义 Amazon Web Services（AWS）区域，您必须上传属于该区域的自定义 Red Hat Enterprise Linux CoreOS（RHCOS）Amazon Machine Image（AMI）。

先决条件

- 已配置了一个 AWS 帐户。
- 已使用所需的 IAM 服务角色创建 Amazon S3 存储桶。
- 将 RHCOS VMDK 文件上传到 Amazon S3。RHCOS VMDK 文件必须是小于或等于您要安装的 OpenShift Container Platform 版本的最高版本。
- 您下载了 AWS CLI 并安装到您的计算机上。请参阅使用捆绑安装程序安装 AWS CLI。

流程

1. 将 AWS 配置集导出为环境变量：

```
$ export AWS_PROFILE=<aws_profile> 1
```

1 拥有 AWS 凭证的 AWS 配置集名称，如 **govcloud**。

2. 将与自定义 AMI 关联的区域导出为环境变量：

```
$ export AWS_DEFAULT_REGION=<aws_region> 1
```

1 AWS 区域，如 **us-gov-east-1**。

3. 将上传至 Amazon S3 的 RHCOS 版本导出为环境变量：

```
$ export RHCOS_VERSION=<version> 1
```

1 RHCOS VMDK 版本，如 **4.6.0**。

4. 将 Amazon S3 存储桶名称导出为环境变量：

```
$ export VMIMPORT_BUCKET_NAME=<s3_bucket_name>
```

5. 创建 **containers.json** 文件并定义 RHCOS VMDK 文件：

```
$ cat <<EOF > containers.json
{
  "Description": "rhcos-${RHCOS_VERSION}-x86_64-aws.x86_64",
  "Format": "vmdk",
  "UserBucket": {
    "S3Bucket": "${VMIMPORT_BUCKET_NAME}",
    "S3Key": "rhcos-${RHCOS_VERSION}-x86_64-aws.x86_64.vmdk"
  }
}
EOF
```

6. 将 RHCOS 磁盘导入为 Amazon EBS 快照：

```
$ aws ec2 import-snapshot --region ${AWS_DEFAULT_REGION} \
    --description "<description>" \ 1
    --disk-container "file://<file_path>/containers.json" 2
```

1 导入 RHCOS 磁盘的描述，如 **rhcos-${RHCOS_VERSION}-x86_64-aws.x86_64**。

2 描述 RHCOS 磁盘的 JSON 文件的文件路径。JSON 文件应包含您的 Amazon S3 存储桶名称和密钥。

7. 检查镜像导入的状态：

```
$ watch -n 5 aws ec2 describe-import-snapshot-tasks --region ${AWS_DEFAULT_REGION}
```

输出示例

```
{
    "ImportSnapshotTasks": [
        {
            "Description": "rhcos-4.6.0-x86_64-aws.x86_64",
            "ImportTaskId": "import-snap-fh6i8uil",
            "SnapshotTaskDetail": {
                "Description": "rhcos-4.6.0-x86_64-aws.x86_64",
                "DiskImageSize": 819056640.0,
                "Format": "VMDK",
                "SnapshotId": "snap-06331325870076318",
                "Status": "completed",
                "UserBucket": {
                    "S3Bucket": "external-images",
                    "S3Key": "rhcos-4.6.0-x86_64-aws.x86_64.vmdk"
                }
            }
        }
    ]
}
```

复制 **SnapshotId** 以注册镜像。

8. 从 RHCOS 快照创建自定义 RHCOS AMI:

```
$ aws ec2 register-image \
  --region ${AWS_DEFAULT_REGION} \
  --architecture x86_64 \ 1
  --description "rhcos-${RHCOS_VERSION}-x86_64-aws.x86_64" \ 2
  --ena-support \
  --name "rhcos-${RHCOS_VERSION}-x86_64-aws.x86_64" \ 3
  --virtualization-type hvm \
  --root-device-name '/dev/xvda' \
  --block-device-mappings 'DeviceName=/dev/xvda,Ebs=
{DeleteOnTermination=true,SnapshotId=<snapshot_ID>}' 4
```

1 RHCOS VMDK 架构类型，如 **x86_64**、**s390x** 或 **ppc64le**。

2 来自导入快照的 **Description**。

3 RHCOS AMI 的名称。

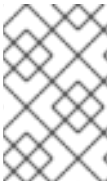
4 导入的快照中的 **SnapshotID**。

如需了解更多有关这些 API 的信息，请参阅 AWS 文档 导入快照 和 创建由 EBS 支持的 AMI。

2.10.12. 在 AWS 中创建 bootstrap 节点

您必须在 Amazon Web Services (AWS) 中创建 bootstrap 节点，以便在 OpenShift Container Platform 集群初始化过程中使用。

您可以使用提供的 CloudFormation 模板和自定义参数文件来创建 AWS 资源堆栈。堆栈代表 OpenShift Container Platform 安装所需的 bootstrap 节点。



注意

如果不使用提供的 CloudFormation 模板来创建 bootstrap 节点，您必须检查提供的信息并手动创建基础架构。如果集群没有正确初始化，您可能需要联系红帽支持并提供您的安装日志。

先决条件

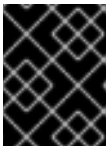
- 已配置了一个 AWS 帐户。
- 您可以通过运行 **aws configure**，将 AWS 密钥和区域添加到本地 AWS 配置集中。
- 已为集群生成 Ignition 配置文件。
- 您在 AWS 中创建并配置了 VPC 及相关子网。
- 您在 AWS 中创建并配置了 DNS、负载均衡器和监听程序。
- 您在 AWS 中创建了集群所需的安全组和角色。

流程

1. 提供一个位置，以便向集群提供 **bootstrap.ign** Ignition 配置文件。此文件位于您的安装目录中。达成此目标的一种方式是在集群区域中创建一个 S3 存储桶，并将 Ignition 配置文件上传到其中。

重要

提供的 CloudFormation 模板假定集群的 Ignition 配置文件由 S3 存储桶提供。如果选择从其他位置提供文件，您必须修改模板。



重要

如果您部署到具有与 AWS SDK 不同的端点，或者您提供自己的自定义端点的区域，则必须为 S3 存储桶使用预签名 URL 而不是 **s3://** 模式。

注意

bootstrap Ignition 配置文件包含 secret，如 X.509 密钥。以下步骤为 S3 存储桶提供基本安全性。若要提供额外的安全性，您可以启用 S3 存储桶策略，仅允许某些用户（如 OpenShift IAM 用户）访问存储桶中包含的对象。您可以完全避开 S3，并从 bootstrap 可访问的任意地址提供 bootstrap Ignition 配置文件。

a. 创建存储桶：

```
$ aws s3 mb s3://<cluster-name>-infra 1
```

1 **<cluster-name>-infra** 是存储桶名称。在创建 **install-config.yaml** 文件时，将 **<cluster-name>** 替换为为集群指定的名称。

b. 将 **bootstrap.ign** Ignition 配置文件上传到存储桶：

```
$ aws s3 cp <installation_directory>/bootstrap.ign s3://<cluster-name>-
infra/bootstrap.ign 1
```

1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

c. 验证文件已经上传：

```
$ aws s3 ls s3://<cluster-name>-infra/
```

输出示例

```
2019-04-03 16:15:16     314878 bootstrap.ign
```

2. 创建一个 JSON 文件，其包含模板所需的参数值：

```
[
 {
  "ParameterKey": "InfrastructureName", 1
  "ParameterValue": "mycluster-<random_string>" 2
 },
 {
  "ParameterKey": "RhcosAmi", 3
  "ParameterValue": "ami-<random_string>" 4
 },
 {
  "ParameterKey": "AllowedBootstrapSshCidr", 5
  "ParameterValue": "0.0.0.0/0" 6
 },
 {
  "ParameterKey": "PublicSubnet", 7
  "ParameterValue": "subnet-<random_string>" 8
 },
 {
  "ParameterKey": "MasterSecurityGroupId", 9
  "ParameterValue": "sg-<random_string>" 10
```

```
  },
  {
    "ParameterKey": "VpcId", (11)
    "ParameterValue": "vpc-<random_string>" (12)
  },
  {
    "ParameterKey": "BootstrapIgnitionLocation", (13)
    "ParameterValue": "s3://<bucket_name>/bootstrap.ign" (14)
  },
  {
    "ParameterKey": "AutoRegisterELB", (15)
    "ParameterValue": "yes" (16)
  },
  {
    "ParameterKey": "RegisterNlbIpTargetsLambdaArn", (17)
    "ParameterValue": "arn:aws:lambda:<region>:<account_number>:function:
<dns_stack_name>-RegisterNlbIpTargets-<random_string>" (18)
  },
  {
    "ParameterKey": "ExternalApiTargetGroupArn", (19)
    "ParameterValue": "arn:aws:elasticloadbalancing:<region>:
<account_number>:targetgroup/<dns_stack_name>-Exter-<random_string>" (20)
  },
  {
    "ParameterKey": "InternalApiTargetGroupArn", (21)
    "ParameterValue": "arn:aws:elasticloadbalancing:<region>:
<account_number>:targetgroup/<dns_stack_name>-Inter-<random_string>" (22)
  },
  {
    "ParameterKey": "InternalServiceTargetGroupArn", (23)
    "ParameterValue": "arn:aws:elasticloadbalancing:<region>:
<account_number>:targetgroup/<dns_stack_name>-Inter-<random_string>" (24)
  }
]
```

1. 您的 Ignition 配置文件中为集群编码的集群基础架构名称。
2. 指定从 Ignition 配置文件元数据中提取的基础架构名称，其格式为 **<cluster-name>-<random-string>**。
3. 用于 bootstrap 节点的当前 Red Hat Enterprise Linux CoreOS (RHCOS) AMI。
4. 指定有效的 **AWS::EC2::Image::Id** 值。
5. 允许通过 SSH 访问 bootstrap 节点的 CIDR 块。
6. 以 **x.x.x.x/16-24** 格式指定 CIDR 块。
7. 与 VPC 关联的公共子网，将 bootstrap 节点启动到其中。
8. 指定 VPC 的 CloudFormation 模板输出的 **PublicSubnetIds** 值。
9. master 安全组 ID（用于注册临时规则）

10 指定安全组和角色的 CloudFormation 模板输出的 **MasterSecurityGroupId** 值。

11 创建的资源将从属于的 VPC。

12 指定 VPC 的 CloudFormation 模板输出的 **VpcId** 值。

13 从中获取 bootstrap Ignition 配置文件的位置。

14 指定 S3 存储桶和文件名，格式为 **s3://<bucket_name>/bootstrap.ign**。

15 是否要注册网络负载均衡器 (NLB) 。

16 指定 **yes** 或 **no**。如果指定 **yes**，您必须提供一个 Lambda Amazon Resource Name (ARN) 值。

17 NLB IP 目标注册 lambda 组的 ARN。

18 指定 DNS 和负载均衡的 CloudFormation 模板输出的 **RegisterNlbIpTargetsLambda** 值。如果将集群部署到 AWS GovCloud 区域，请使用 **arn:aws-us-gov**。

19 外部 API 负载均衡器目标组的 ARN。

20 指定 DNS 和负载均衡的 CloudFormation 模板输出的 **ExternalApiTargetGroupArn** 值。如果将集群部署到 AWS GovCloud 区域，请使用 **arn:aws-us-gov**。

21 内部 API 负载均衡器目标组群的 ARN。

22 指定 DNS 和负载均衡的 CloudFormation 模板输出的 **InternalApiTargetGroupArn** 值。如果将集群部署到 AWS GovCloud 区域，请使用 **arn:aws-us-gov**。

23 内部服务负载均衡器目标组群的 ARN。

24 指定 DNS 和负载均衡的 CloudFormation 模板输出的 **InternalServiceTargetGroupArn** 值。如果将集群部署到 AWS GovCloud 区域，请使用 **arn:aws-us-gov**。

3. 复制本主题的 **Bootstrap 机器的 CloudFormation 模板**部分中的模板，并将它以 YAML 文件形式保存到计算机上。此模板描述了集群所需的 bootstrap 机器。

4. 启动 CloudFormation 模板，以创建代表 bootstrap 节点的 AWS 资源堆栈：

重要

您必须在一行内输入命令。

```
$ aws cloudformation create-stack --stack-name <name> 1
    --template-body file://<template>.yaml 2
    --parameters file://<parameters>.json 3
    --capabilities CAPABILITY_NAMED_IAM 4
```

1 **<name>** 是 CloudFormation 堆栈的名称，如 **cluster-bootstrap**。如果您删除集群，则需要此堆栈的名称。

2 **<template>** 是您保存的 CloudFormation 模板 YAML 文件的相对路径和名称。

3. **<parameters>** 是 CloudFormation 参数 JSON 文件的相对路径和名称。
4. 您必须明确声明 **CAPABILITY_NAMED_IAM** 功能，因为提供的模板会创建一些 **AWS::IAM::Role** 和 **AWS::IAM::InstanceProfile** 资源。

输出示例

```
arn:aws:cloudformation:us-east-1:269333783861:stack/cluster-bootstrap/12944486-2add-11eb-9dee-12dace8e3a83
```

5. 确认模板组件已存在：

```
$ aws cloudformation describe-stacks --stack-name <name>
```

在 **StackStatus** 显示 **CREATE_COMPLETE** 后，输出会显示以下参数的值。您必须将这些参数值提供给您在创建集群时要运行的其他 CloudFormation 模板：

BootstrapInstanceId	bootstrap 实例 ID。
BootstrapPublicIp	bootstrap 节点公共 IP 地址。
BootstrapPrivateIp	bootstrap 节点专用 IP 地址。

2.10.12.1. bootstrap 机器的 CloudFormation 模板

您可以使用以下 CloudFormation 模板来部署 OpenShift Container Platform 集群所需的 bootstrap 机器。

例 2.28. bootstrap 机器的 CloudFormation 模板

```
AWSTemplateFormatVersion: 2010-09-09
Description: Template for OpenShift Cluster Bootstrap (EC2 Instance, Security Groups and IAM)

Parameters:
  InfrastructureName:
    AllowedPattern: ^([a-zA-Z][a-zA-Z0-9\-]{0,26})$
    MaxLength: 27
    MinLength: 1
    ConstraintDescription: Infrastructure name must be alphanumeric, start with a letter, and have a maximum of 27 characters.
    Description: A short, unique cluster ID used to tag cloud resources and identify items owned or used by the cluster.
    Type: String
  RhcosAmi:
    Description: Current Red Hat Enterprise Linux CoreOS AMI to use for bootstrap.
    Type: AWS::EC2::Image::Id
  AllowedBootstrapSshCidr:
    AllowedPattern: ^(([0-9]|[1-9][0-9]|1[0-9]{2}|2[0-4][0-9]|25[0-5])\.){3}([0-9]|[1-9][0-9]|1[0-9]{2}|2[0-4][0-9]|25[0-5])(\/([0-9]|1[0-9]|2[0-9]|3[0-2]))$
```

```
    ConstraintDescription: CIDR block parameter must be in the form x.x.x.x/0-32.
    Default: 0.0.0.0/0
    Description: CIDR block to allow SSH access to the bootstrap node.
    Type: String
  PublicSubnet:
    Description: The public subnet to launch the bootstrap node into.
    Type: AWS::EC2::Subnet::Id
  MasterSecurityGroupId:
    Description: The master security group ID for registering temporary rules.
    Type: AWS::EC2::SecurityGroup::Id
  VpcId:
    Description: The VPC-scoped resources will belong to this VPC.
    Type: AWS::EC2::VPC::Id
  BootstrapIgnitionLocation:
    Default: s3://my-s3-bucket/bootstrap.ign
    Description: Ignition config file location.
    Type: String
  AutoRegisterELB:
    Default: "yes"
    AllowedValues:
    - "yes"
    - "no"
    Description: Do you want to invoke NLB registration, which requires a Lambda ARN parameter?
    Type: String
  RegisterNlbIpTargetsLambdaArn:
    Description: ARN for NLB IP target registration lambda.
    Type: String
  ExternalApiTargetGroupArn:
    Description: ARN for external API load balancer target group.
    Type: String
  InternalApiTargetGroupArn:
    Description: ARN for internal API load balancer target group.
    Type: String
  InternalServiceTargetGroupArn:
    Description: ARN for internal service load balancer target group.
    Type: String

Metadata:
  AWS::CloudFormation::Interface:
    ParameterGroups:
    - Label:
        default: "Cluster Information"
      Parameters:
      - InfrastructureName
    - Label:
        default: "Host Information"
      Parameters:
      - RhcosAmi
      - BootstrapIgnitionLocation
      - MasterSecurityGroupId
    - Label:
        default: "Network Configuration"
      Parameters:
      - VpcId
      - AllowedBootstrapSshCidr
      - PublicSubnet
```

```
    - Label:
        default: "Load Balancer Automation"
      Parameters:
      - AutoRegisterELB
      - RegisterNlbIpTargetsLambdaArn
      - ExternalApiTargetGroupArn
      - InternalApiTargetGroupArn
      - InternalServiceTargetGroupArn
    ParameterLabels:
      InfrastructureName:
        default: "Infrastructure Name"
      VpcId:
        default: "VPC ID"
      AllowedBootstrapSshCidr:
        default: "Allowed SSH Source"
      PublicSubnet:
        default: "Public Subnet"
      RhcosAmi:
        default: "Red Hat Enterprise Linux CoreOS AMI ID"
      BootstrapIgnitionLocation:
        default: "Bootstrap Ignition Source"
      MasterSecurityGroupId:
        default: "Master Security Group ID"
      AutoRegisterELB:
        default: "Use Provided ELB Automation"

Conditions:
  DoRegistration: !Equals ["yes", !Ref AutoRegisterELB]

Resources:
  BootstrapIamRole:
    Type: AWS::IAM::Role
    Properties:
      AssumeRolePolicyDocument:
        Version: "2012-10-17"
        Statement:
        - Effect: "Allow"
          Principal:
            Service:
            - "ec2.amazonaws.com"
          Action:
          - "sts:AssumeRole"
      Path: "/"
      Policies:
      - PolicyName: !Join ["-", [!Ref InfrastructureName, "bootstrap", "policy"]]
        PolicyDocument:
          Version: "2012-10-17"
          Statement:
          - Effect: "Allow"
            Action: "ec2:Describe*"
            Resource: "*"
          - Effect: "Allow"
            Action: "ec2:AttachVolume"
            Resource: "*"
          - Effect: "Allow"
            Action: "ec2:DetachVolume"
```

```
          Resource: "*"
        - Effect: "Allow"
          Action: "s3:GetObject"
          Resource: "*"

BootstrapInstanceProfile:
  Type: "AWS::IAM::InstanceProfile"
  Properties:
    Path: "/"
    Roles:
    - Ref: "BootstrapIamRole"

BootstrapSecurityGroup:
  Type: AWS::EC2::SecurityGroup
  Properties:
    GroupDescription: Cluster Bootstrap Security Group
    SecurityGroupIngress:
    - IpProtocol: tcp
      FromPort: 22
      ToPort: 22
      CidrIp: !Ref AllowedBootstrapSshCidr
    - IpProtocol: tcp
      ToPort: 19531
      FromPort: 19531
      CidrIp: 0.0.0.0/0
    VpcId: !Ref VpcId

BootstrapInstance:
  Type: AWS::EC2::Instance
  Properties:
    ImageId: !Ref RhcosAmi
    IamInstanceProfile: !Ref BootstrapInstanceProfile
    InstanceType: "i3.large"
    NetworkInterfaces:
    - AssociatePublicIpAddress: "true"
      DeviceIndex: "0"
      GroupSet:
      - !Ref "BootstrapSecurityGroup"
      - !Ref "MasterSecurityGroupId"
      SubnetId: !Ref "PublicSubnet"
    UserData:
      Fn::Base64: !Sub
      - '{"ignition":{"config":{"replace":{"source":"${S3Loc}"}},"version":"3.1.0"}}'
      - {
        S3Loc: !Ref BootstrapIgnitionLocation
      }

RegisterBootstrapApiTarget:
  Condition: DoRegistration
  Type: Custom::NLBRegister
  Properties:
    ServiceToken: !Ref RegisterNlbIpTargetsLambdaArn
    TargetArn: !Ref ExternalApiTargetGroupArn
    TargetIp: !GetAtt BootstrapInstance.PrivateIp

RegisterBootstrapInternalApiTarget:
```

```
      Condition: DoRegistration
      Type: Custom::NLBRegister
      Properties:
        ServiceToken: !Ref RegisterNlbIpTargetsLambdaArn
        TargetArn: !Ref InternalApiTargetGroupArn
        TargetIp: !GetAtt BootstrapInstance.PrivateIp

    RegisterBootstrapInternalServiceTarget:
      Condition: DoRegistration
      Type: Custom::NLBRegister
      Properties:
        ServiceToken: !Ref RegisterNlbIpTargetsLambdaArn
        TargetArn: !Ref InternalServiceTargetGroupArn
        TargetIp: !GetAtt BootstrapInstance.PrivateIp

  Outputs:
    BootstrapInstanceId:
      Description: Bootstrap Instance ID.
      Value: !Ref BootstrapInstance

    BootstrapPublicIp:
      Description: The bootstrap node public IP address.
      Value: !GetAtt BootstrapInstance.PublicIp

    BootstrapPrivateIp:
      Description: The bootstrap node private IP address.
      Value: !GetAtt BootstrapInstance.PrivateIp
```

其他资源

- 您可以通过导航 AWS CloudFormation 控制台来查看您创建的 CloudFormation 堆栈的详情。
- 如需有关 AWS 区的 Red Hat Enterprise Linux CoreOS（RHCOS）AMI 的详细信息，请参阅 AWS 基础架构的 RHCOS AMI。

2.10.13. 在 AWS 中创建 control plane 机器

您必须在集群要使用的 Amazon Web Services（AWS）中创建 control plane 机器。

您可以使用提供的 CloudFormation 模板和自定义参数文件，创建代表 control plane 节点的 AWS 资源堆栈。

重要

CloudFormation 模板会创建一个堆栈，它代表三个 control plane 节点。

注意

如果不使用提供的 CloudFormation 模板来创建 control plane 节点，您必须检查提供的信息并手动创建基础架构。如果集群没有正确初始化，您可能需要联系红帽支持并提供您的安装日志。

先决条件

- 已配置了一个 AWS 帐户。
- 您可以通过运行 **aws configure**，将 AWS 密钥和区域添加到本地 AWS 配置集中。
- 已为集群生成 Ignition 配置文件。
- 您在 AWS 中创建并配置了 VPC 及相关子网。
- 您在 AWS 中创建并配置了 DNS、负载均衡器和监听程序。
- 您在 AWS 中创建了集群所需的安全组和角色。
- 已创建 bootstrap 机器。

流程

1. 创建一个 JSON 文件，其包含模板所需的参数值：

```
[
  {
    "ParameterKey": "InfrastructureName", 1
    "ParameterValue": "mycluster-<random_string>" 2
  },
  {
    "ParameterKey": "RhcosAmi", 3
    "ParameterValue": "ami-<random_string>" 4
  },
  {
    "ParameterKey": "AutoRegisterDNS", 5
    "ParameterValue": "yes" 6
  },
  {
    "ParameterKey": "PrivateHostedZoneId", 7
    "ParameterValue": "<random_string>" 8
  },
  {
    "ParameterKey": "PrivateHostedZoneName", 9
    "ParameterValue": "mycluster.example.com" 10
  },
  {
    "ParameterKey": "Master0Subnet", 11
    "ParameterValue": "subnet-<random_string>" 12
  },
  {
    "ParameterKey": "Master1Subnet", 13
    "ParameterValue": "subnet-<random_string>" 14
  },
  {
    "ParameterKey": "Master2Subnet", 15
    "ParameterValue": "subnet-<random_string>" 16
  },
  {
    "ParameterKey": "MasterSecurityGroupId", 17
    "ParameterValue": "sg-<random_string>" 18
```

```
  },
  {
    "ParameterKey": "IgnitionLocation", 19
    "ParameterValue": "https://api-int.<cluster_name>.<domain_name>:22623/config/master"
20
  },
  {
    "ParameterKey": "CertificateAuthorities", 21
    "ParameterValue": "data:text/plain;charset=utf-8;base64,ABC...xYz==" 22
  },
  {
    "ParameterKey": "MasterInstanceProfileName", 23
    "ParameterValue": "<roles_stack>-MasterInstanceProfile-<random_string>" 24
  },
  {
    "ParameterKey": "MasterInstanceType", 25
    "ParameterValue": "m4.xlarge" 26
  },
  {
    "ParameterKey": "AutoRegisterELB", 27
    "ParameterValue": "yes" 28
  },
  {
    "ParameterKey": "RegisterNlbIpTargetsLambdaArn", 29
    "ParameterValue": "arn:aws:lambda:<region>:<account_number>:function:
<dns_stack_name>-RegisterNlbIpTargets-<random_string>" 30
  },
  {
    "ParameterKey": "ExternalApiTargetGroupArn", 31
    "ParameterValue": "arn:aws:elasticloadbalancing:<region>:
<account_number>:targetgroup/<dns_stack_name>-Exter-<random_string>" 32
  },
  {
    "ParameterKey": "InternalApiTargetGroupArn", 33
    "ParameterValue": "arn:aws:elasticloadbalancing:<region>:
<account_number>:targetgroup/<dns_stack_name>-Inter-<random_string>" 34
  },
  {
    "ParameterKey": "InternalServiceTargetGroupArn", 35
    "ParameterValue": "arn:aws:elasticloadbalancing:<region>:
<account_number>:targetgroup/<dns_stack_name>-Inter-<random_string>" 36
  }
]
```

1. 您的 Ignition 配置文件中为集群编码的集群基础架构名称。
2. 指定从 Ignition 配置文件元数据中提取的基础架构名称，其格式为 **<cluster-name>-<random-string>**。
3. 用于 control plane 机器的当前 Red Hat Enterprise Linux CoreOS (RHCOS) AMI。
4. 指定 **AWS::EC2::Image::Id** 值。
5. 是否要执行 DNS etcd 注册。

6 指定 **yes** 或 **no**。如果指定 **yes**，您必须提供托管区信息。

7 用来注册 etcd 目标的 Route 53 专用区 ID。

8 指定 DNS 和负载均衡的 CloudFormation 模板输出的 **PrivateHostedZoneId** 值。

9 用来注册目标的 Route 53 区。

10 指定 **<cluster_name>.<domain_name>**，其中 **<domain_name>** 是您为集群生成 **install-config.yaml** 文件时所用的 Route 53 基域。请勿包含 AWS 控制台中显示的结尾句点 (.)。

11 13 15 在其中启动 control plane 机器的子网，最好是专用子网。

12 14 16 从 DNS 和负载均衡的 CloudFormation 模板输出的 **PrivateSubnets** 值指定子网。

17 与 control plane 节点（也称为 master 节点）关联的 master 安全组 ID。

18 指定安全组和角色的 CloudFormation 模板输出的 **MasterSecurityGroupId** 值。

19 从中获取 control plane Ignition 配置文件的位置。

20 指定生成的 Ignition 配置文件的位置，**https://api-int.<cluster_name>.<domain_name>:22623/config/master**。

21 要使用的 base64 编码证书颁发机构字符串。

22 指定安装目录中 **master.ign** 文件中的值。这个值是一个长字符串，格式为 **data:text/plain;charset=utf-8;base64,ABC...xYz==**。

23 与 control plane 节点关联的 IAM 配置集。

24 指定安全组和角色的 CloudFormation 模板输出的 **MasterInstanceProfile** 参数值。

25 用于 control plane 机器的 AWS 实例类型。

26 允许的值：

- **m4.xlarge**
- **m4.2xlarge**
- **m4.4xlarge**
- **m4.8xlarge**
- **m4.10xlarge**
- **m4.16xlarge**
- **m5.xlarge**
- **m5.2xlarge**
- **m5.4xlarge**
- **m5.8xlarge**
- **m5.10xlarge**

- **m5.16xlarge**
- **m6i.xlarge**
- **c4.2xlarge**
- **c4.4xlarge**
- **c4.8xlarge**
- **r4.xlarge**
- **r4.2xlarge**
- **r4.4xlarge**
- **r4.8xlarge**
- **r4.16xlarge**

重要

如果您的区域中没有 **m4** 实例类型，例如 **eu-west-3**，请改为指定 **m5** 类型，如 **m5.xlarge**。

27 是否要注册网络负载均衡器 (NLB) 。

28 指定 **yes** 或 **no**。如果指定 **yes**，您必须提供一个 Lambda Amazon Resource Name (ARN) 值。

29 NLB IP 目标注册 lambda 组的 ARN。

30 指定 DNS 和负载均衡的 CloudFormation 模板输出的 **RegisterNlbIpTargetsLambda** 值。如果将集群部署到 AWS GovCloud 区域，请使用 **arn:aws-us-gov**。

31 外部 API 负载均衡器目标组的 ARN。

32 指定 DNS 和负载均衡的 CloudFormation 模板输出的 **ExternalApiTargetGroupArn** 值。如果将集群部署到 AWS GovCloud 区域，请使用 **arn:aws-us-gov**。

33 内部 API 负载均衡器目标组群的 ARN。

34 指定 DNS 和负载均衡的 CloudFormation 模板输出的 **InternalApiTargetGroupArn** 值。如果将集群部署到 AWS GovCloud 区域，请使用 **arn:aws-us-gov**。

35 内部服务负载均衡器目标组群的 ARN。

36 指定 DNS 和负载均衡的 CloudFormation 模板输出的 **InternalServiceTargetGroupArn** 值。如果将集群部署到 AWS GovCloud 区域，请使用 **arn:aws-us-gov**。

2. 复制**control plane 机器的 CloudFormation 模板**一节中的模板，并将它以 YAML 文件形式保存到计算机上。此模板描述了集群所需的 control plane 机器。
3. 如果您将 **m5** 实例类型指定为 **MasterInstanceType** 的值，请将该实例类型添加到 CloudFormation 模板中的 **MasterInstanceType.AllowedValues** 参数。

4. 启动 CloudFormation 模板，以创建代表 control plane 节点的 AWS 资源堆栈：

重要

您必须在一行内输入命令。

```
$ aws cloudformation create-stack --stack-name <name> 1
    --template-body file://<template>.yaml 2
    --parameters file://<parameters>.json 3
```

1. **<name>** 是 CloudFormation 堆栈的名称，如 **cluster-control-plane**。如果您删除集群，则需要此堆栈的名称。
2. **<template>** 是您保存的 CloudFormation 模板 YAML 文件的相对路径和名称。
3. **<parameters>** 是 CloudFormation 参数 JSON 文件的相对路径和名称。

输出示例

```
arn:aws:cloudformation:us-east-1:269333783861:stack/cluster-control-plane/21c7e2b0-2ee2-11eb-c6f6-0aa34627df4b
```

注意

CloudFormation 模板会创建一个堆栈，它代表三个 control plane 节点。

5. 确认模板组件已存在：

```
$ aws cloudformation describe-stacks --stack-name <name>
```

2.10.13.1. control plane 机器的 CloudFormation 模板

您可以使用以下 CloudFormation 模板来部署 OpenShift Container Platform 集群所需的 control plane 机器。

例 2.29. control plane 机器的 CloudFormation 模板

```
AWSTemplateFormatVersion: 2010-09-09
Description: Template for OpenShift Cluster Node Launch (EC2 master instances)

Parameters:
  InfrastructureName:
    AllowedPattern: ^([a-zA-Z][a-zA-Z0-9\-]{0,26})$
    MaxLength: 27
    MinLength: 1
    ConstraintDescription: Infrastructure name must be alphanumeric, start with a letter, and have a
maximum of 27 characters.
    Description: A short, unique cluster ID used to tag nodes for the kubelet cloud provider.
    Type: String
  RhcosAmi:
    Description: Current Red Hat Enterprise Linux CoreOS AMI to use for bootstrap.
```

```
    Type: AWS::EC2::Image::Id
  AutoRegisterDNS:
    Default: "yes"
    AllowedValues:
    - "yes"
    - "no"
    Description: Do you want to invoke DNS etcd registration, which requires Hosted Zone
information?
    Type: String
  PrivateHostedZoneId:
    Description: The Route53 private zone ID to register the etcd targets with, such as
Z21IXYZABCZ2A4.
    Type: String
  PrivateHostedZoneName:
    Description: The Route53 zone to register the targets with, such as cluster.example.com. Omit
the trailing period.
    Type: String
  Master0Subnet:
    Description: The subnets, recommend private, to launch the master nodes into.
    Type: AWS::EC2::Subnet::Id
  Master1Subnet:
    Description: The subnets, recommend private, to launch the master nodes into.
    Type: AWS::EC2::Subnet::Id
  Master2Subnet:
    Description: The subnets, recommend private, to launch the master nodes into.
    Type: AWS::EC2::Subnet::Id
  MasterSecurityGroupId:
    Description: The master security group ID to associate with master nodes.
    Type: AWS::EC2::SecurityGroup::Id
  IgnitionLocation:
    Default: https://api-int.$CLUSTER_NAME.$DOMAIN:22623/config/master
    Description: Ignition config file location.
    Type: String
  CertificateAuthorities:
    Default: data:text/plain;charset=utf-8;base64,ABC...xYz==
    Description: Base64 encoded certificate authority string to use.
    Type: String
  MasterInstanceProfileName:
    Description: IAM profile to associate with master nodes.
    Type: String
  MasterInstanceType:
    Default: m5.xlarge
    Type: String
    AllowedValues:
    - "m4.xlarge"
    - "m4.2xlarge"
    - "m4.4xlarge"
    - "m4.10xlarge"
    - "m4.16xlarge"
    - "m5.xlarge"
    - "m5.2xlarge"
    - "m5.4xlarge"
    - "m5.8xlarge"
    - "m5.12xlarge"
    - "m5.16xlarge"
    - "m5a.xlarge"
```

```
      - "m5a.2xlarge"
      - "m5a.4xlarge"
      - "m5a.8xlarge"
      - "m5a.10xlarge"
      - "m5a.16xlarge"
      - "c4.2xlarge"
      - "c4.4xlarge"
      - "c4.8xlarge"
      - "c5.2xlarge"
      - "c5.4xlarge"
      - "c5.9xlarge"
      - "c5.12xlarge"
      - "c5.18xlarge"
      - "c5.24xlarge"
      - "c5a.2xlarge"
      - "c5a.4xlarge"
      - "c5a.8xlarge"
      - "c5a.12xlarge"
      - "c5a.16xlarge"
      - "c5a.24xlarge"
      - "r4.xlarge"
      - "r4.2xlarge"
      - "r4.4xlarge"
      - "r4.8xlarge"
      - "r4.16xlarge"
      - "r5.xlarge"
      - "r5.2xlarge"
      - "r5.4xlarge"
      - "r5.8xlarge"
      - "r5.12xlarge"
      - "r5.16xlarge"
      - "r5.24xlarge"
      - "r5a.xlarge"
      - "r5a.2xlarge"
      - "r5a.4xlarge"
      - "r5a.8xlarge"
      - "r5a.12xlarge"
      - "r5a.16xlarge"
      - "r5a.24xlarge"

    AutoRegisterELB:
      Default: "yes"
      AllowedValues:
      - "yes"
      - "no"
      Description: Do you want to invoke NLB registration, which requires a Lambda ARN parameter?
      Type: String
    RegisterNlbIpTargetsLambdaArn:
      Description: ARN for NLB IP target registration lambda. Supply the value from the cluster
infrastructure or select "no" for AutoRegisterELB.
      Type: String
    ExternalApiTargetGroupArn:
      Description: ARN for external API load balancer target group. Supply the value from the cluster
infrastructure or select "no" for AutoRegisterELB.
      Type: String
    InternalApiTargetGroupArn:
```

```
    Description: ARN for internal API load balancer target group. Supply the value from the cluster
infrastructure or select "no" for AutoRegisterELB.
    Type: String
  InternalServiceTargetGroupArn:
    Description: ARN for internal service load balancer target group. Supply the value from the
cluster infrastructure or select "no" for AutoRegisterELB.
    Type: String

Metadata:
  AWS::CloudFormation::Interface:
    ParameterGroups:
    - Label:
        default: "Cluster Information"
      Parameters:
      - InfrastructureName
    - Label:
        default: "Host Information"
      Parameters:
      - MasterInstanceType
      - RhcosAmi
      - IgnitionLocation
      - CertificateAuthorities
      - MasterSecurityGroupId
      - MasterInstanceProfileName
    - Label:
        default: "Network Configuration"
      Parameters:
      - VpcId
      - AllowedBootstrapSshCidr
      - Master0Subnet
      - Master1Subnet
      - Master2Subnet
    - Label:
        default: "DNS"
      Parameters:
      - AutoRegisterDNS
      - PrivateHostedZoneName
      - PrivateHostedZoneId
    - Label:
        default: "Load Balancer Automation"
      Parameters:
      - AutoRegisterELB
      - RegisterNlbIpTargetsLambdaArn
      - ExternalApiTargetGroupArn
      - InternalApiTargetGroupArn
      - InternalServiceTargetGroupArn
    ParameterLabels:
      InfrastructureName:
        default: "Infrastructure Name"
      VpcId:
        default: "VPC ID"
      Master0Subnet:
        default: "Master-0 Subnet"
      Master1Subnet:
        default: "Master-1 Subnet"
      Master2Subnet:
```

```
      default: "Master-2 Subnet"
    MasterInstanceType:
      default: "Master Instance Type"
    MasterInstanceProfileName:
      default: "Master Instance Profile Name"
    RhcosAmi:
      default: "Red Hat Enterprise Linux CoreOS AMI ID"
    BootstrapIgnitionLocation:
      default: "Master Ignition Source"
    CertificateAuthorities:
      default: "Ignition CA String"
    MasterSecurityGroupId:
      default: "Master Security Group ID"
    AutoRegisterDNS:
      default: "Use Provided DNS Automation"
    AutoRegisterELB:
      default: "Use Provided ELB Automation"
    PrivateHostedZoneName:
      default: "Private Hosted Zone Name"
    PrivateHostedZoneId:
      default: "Private Hosted Zone ID"

Conditions:
  DoRegistration: !Equals ["yes", !Ref AutoRegisterELB]
  DoDns: !Equals ["yes", !Ref AutoRegisterDNS]

Resources:
  Master0:
    Type: AWS::EC2::Instance
    Properties:
      ImageId: !Ref RhcosAmi
      BlockDeviceMappings:
      - DeviceName: /dev/xvda
        Ebs:
          VolumeSize: "120"
          VolumeType: "gp2"
      IamInstanceProfile: !Ref MasterInstanceProfileName
      InstanceType: !Ref MasterInstanceType
      NetworkInterfaces:
      - AssociatePublicIpAddress: "false"
        DeviceIndex: "0"
        GroupSet:
        - !Ref "MasterSecurityGroupId"
        SubnetId: !Ref "Master0Subnet"
      UserData:
        Fn::Base64: !Sub
        - '{"ignition":{"config":{"merge":[{"source":"${SOURCE}"}]},"security":{"tls":
{"certificateAuthorities":[{"source":"${CA_BUNDLE}"}]}},"version":"3.1.0"}}'
        - {
          SOURCE: !Ref IgnitionLocation,
          CA_BUNDLE: !Ref CertificateAuthorities,
        }
      Tags:
      - Key: !Join ["", ["kubernetes.io/cluster/", !Ref InfrastructureName]]
        Value: "shared"
```

```
  RegisterMaster0:
    Condition: DoRegistration
    Type: Custom::NLBRegister
    Properties:
      ServiceToken: !Ref RegisterNlbIpTargetsLambdaArn
      TargetArn: !Ref ExternalApiTargetGroupArn
      TargetIp: !GetAtt Master0.PrivateIp

  RegisterMaster0InternalApiTarget:
    Condition: DoRegistration
    Type: Custom::NLBRegister
    Properties:
      ServiceToken: !Ref RegisterNlbIpTargetsLambdaArn
      TargetArn: !Ref InternalApiTargetGroupArn
      TargetIp: !GetAtt Master0.PrivateIp

  RegisterMaster0InternalServiceTarget:
    Condition: DoRegistration
    Type: Custom::NLBRegister
    Properties:
      ServiceToken: !Ref RegisterNlbIpTargetsLambdaArn
      TargetArn: !Ref InternalServiceTargetGroupArn
      TargetIp: !GetAtt Master0.PrivateIp

  Master1:
    Type: AWS::EC2::Instance
    Properties:
      ImageId: !Ref RhcosAmi
      BlockDeviceMappings:
      - DeviceName: /dev/xvda
        Ebs:
          VolumeSize: "120"
          VolumeType: "gp2"
      IamInstanceProfile: !Ref MasterInstanceProfileName
      InstanceType: !Ref MasterInstanceType
      NetworkInterfaces:
      - AssociatePublicIpAddress: "false"
        DeviceIndex: "0"
        GroupSet:
        - !Ref "MasterSecurityGroupId"
        SubnetId: !Ref "Master1Subnet"
      UserData:
        Fn::Base64: !Sub
        - '{"ignition":{"config":{"merge":[{"source":"${SOURCE}"}]},"security":{"tls":
{"certificateAuthorities":[{"source":"${CA_BUNDLE}"}]}},"version":"3.1.0"}}'
        - {
          SOURCE: !Ref IgnitionLocation,
          CA_BUNDLE: !Ref CertificateAuthorities,
        }
      Tags:
      - Key: !Join ["", ["kubernetes.io/cluster/", !Ref InfrastructureName]]
        Value: "shared"

  RegisterMaster1:
    Condition: DoRegistration
    Type: Custom::NLBRegister
```

```
    Properties:
      ServiceToken: !Ref RegisterNlbIpTargetsLambdaArn
      TargetArn: !Ref ExternalApiTargetGroupArn
      TargetIp: !GetAtt Master1.PrivateIp

  RegisterMaster1InternalApiTarget:
    Condition: DoRegistration
    Type: Custom::NLBRegister
    Properties:
      ServiceToken: !Ref RegisterNlbIpTargetsLambdaArn
      TargetArn: !Ref InternalApiTargetGroupArn
      TargetIp: !GetAtt Master1.PrivateIp

  RegisterMaster1InternalServiceTarget:
    Condition: DoRegistration
    Type: Custom::NLBRegister
    Properties:
      ServiceToken: !Ref RegisterNlbIpTargetsLambdaArn
      TargetArn: !Ref InternalServiceTargetGroupArn
      TargetIp: !GetAtt Master1.PrivateIp

  Master2:
    Type: AWS::EC2::Instance
    Properties:
      ImageId: !Ref RhcosAmi
      BlockDeviceMappings:
      - DeviceName: /dev/xvda
        Ebs:
          VolumeSize: "120"
          VolumeType: "gp2"
      IamInstanceProfile: !Ref MasterInstanceProfileName
      InstanceType: !Ref MasterInstanceType
      NetworkInterfaces:
      - AssociatePublicIpAddress: "false"
        DeviceIndex: "0"
        GroupSet:
        - !Ref "MasterSecurityGroupId"
        SubnetId: !Ref "Master2Subnet"
      UserData:
        Fn::Base64: !Sub
        - '{"ignition":{"config":{"merge":[{"source":"${SOURCE}"}]},"security":{"tls":
{"certificateAuthorities":[{"source":"${CA_BUNDLE}"}]}},"version":"3.1.0"}}'
        - {
          SOURCE: !Ref IgnitionLocation,
          CA_BUNDLE: !Ref CertificateAuthorities,
        }
      Tags:
      - Key: !Join ["", ["kubernetes.io/cluster/", !Ref InfrastructureName]]
        Value: "shared"

  RegisterMaster2:
    Condition: DoRegistration
    Type: Custom::NLBRegister
    Properties:
      ServiceToken: !Ref RegisterNlbIpTargetsLambdaArn
      TargetArn: !Ref ExternalApiTargetGroupArn
```

```
      TargetIp: !GetAtt Master2.PrivateIp

  RegisterMaster2InternalApiTarget:
    Condition: DoRegistration
    Type: Custom::NLBRegister
    Properties:
      ServiceToken: !Ref RegisterNlbIpTargetsLambdaArn
      TargetArn: !Ref InternalApiTargetGroupArn
      TargetIp: !GetAtt Master2.PrivateIp

  RegisterMaster2InternalServiceTarget:
    Condition: DoRegistration
    Type: Custom::NLBRegister
    Properties:
      ServiceToken: !Ref RegisterNlbIpTargetsLambdaArn
      TargetArn: !Ref InternalServiceTargetGroupArn
      TargetIp: !GetAtt Master2.PrivateIp

  EtcdSrvRecords:
    Condition: DoDns
    Type: AWS::Route53::RecordSet
    Properties:
      HostedZoneId: !Ref PrivateHostedZoneId
      Name: !Join [".", ["_etcd-server-ssl._tcp", !Ref PrivateHostedZoneName]]
      ResourceRecords:
      - !Join [
        " ",
        ["0 10 2380", !Join [".", ["etcd-0", !Ref PrivateHostedZoneName]]],
      ]
      - !Join [
        " ",
        ["0 10 2380", !Join [".", ["etcd-1", !Ref PrivateHostedZoneName]]],
      ]
      - !Join [
        " ",
        ["0 10 2380", !Join [".", ["etcd-2", !Ref PrivateHostedZoneName]]],
      ]
      TTL: 60
      Type: SRV

  Etcd0Record:
    Condition: DoDns
    Type: AWS::Route53::RecordSet
    Properties:
      HostedZoneId: !Ref PrivateHostedZoneId
      Name: !Join [".", ["etcd-0", !Ref PrivateHostedZoneName]]
      ResourceRecords:
      - !GetAtt Master0.PrivateIp
      TTL: 60
      Type: A

  Etcd1Record:
    Condition: DoDns
    Type: AWS::Route53::RecordSet
    Properties:
      HostedZoneId: !Ref PrivateHostedZoneId
```

```
      Name: !Join ["", ["etcd-1", !Ref PrivateHostedZoneName]]
      ResourceRecords:
      - !GetAtt Master1.PrivateIp
      TTL: 60
      Type: A

  Etcd2Record:
    Condition: DoDns
    Type: AWS::Route53::RecordSet
    Properties:
      HostedZoneId: !Ref PrivateHostedZoneId
      Name: !Join ["", ["etcd-2", !Ref PrivateHostedZoneName]]
      ResourceRecords:
      - !GetAtt Master2.PrivateIp
      TTL: 60
      Type: A

Outputs:
  PrivateIPs:
    Description: The control-plane node private IP addresses.
    Value:
      !Join [
        ",",
        [!GetAtt Master0.PrivateIp, !GetAtt Master1.PrivateIp, !GetAtt Master2.PrivateIp]
      ]
```

其他资源

- 您可以通过导航 AWS CloudFormation 控制台来查看您创建的 CloudFormation 堆栈的详情。

2.10.14. 在 AWS 中创建 worker 节点

您可以在 Amazon Web Services (AWS) 中创建 worker 节点，供集群使用。

您可以使用提供的 CloudFormation 模板和自定义参数文件创建代表 worker 节点的 AWS 资源堆栈。

重要

CloudFormation 模板会创建一个堆栈，它代表一个 worker 节点。您必须为每个 worker 节点创建一个堆栈。

注意

如果不使用提供的 CloudFormation 模板来创建 worker 节点，您必须检查提供的信息并手动创建基础架构。如果集群没有正确初始化，您可能需要联系红帽支持并提供您的安装日志。

先决条件

- 已配置了一个 AWS 帐户。
- 您可以通过运行 **aws configure**，将 AWS 密钥和区域添加到本地 AWS 配置集中。

- 已为集群生成 Ignition 配置文件。
- 您在 AWS 中创建并配置了 VPC 及相关子网。
- 您在 AWS 中创建并配置了 DNS、负载均衡器和监听程序。
- 您在 AWS 中创建了集群所需的安全组和角色。
- 已创建 bootstrap 机器。
- 已创建 control plane 机器。

流程

1. 创建一个 JSON 文件，其包含 CloudFormation 模板需要的参数值：

```
[
  {
    "ParameterKey": "InfrastructureName", 1
    "ParameterValue": "mycluster-<random_string>" 2
  },
  {
    "ParameterKey": "RhcosAmi", 3
    "ParameterValue": "ami-<random_string>" 4
  },
  {
    "ParameterKey": "Subnet", 5
    "ParameterValue": "subnet-<random_string>" 6
  },
  {
    "ParameterKey": "WorkerSecurityGroupId", 7
    "ParameterValue": "sg-<random_string>" 8
  },
  {
    "ParameterKey": "IgnitionLocation", 9
    "ParameterValue": "https://api-int.<cluster_name>.<domain_name>:22623/config/worker"
10
  },
  {
    "ParameterKey": "CertificateAuthorities", 11
    "ParameterValue": "" 12
  },
  {
    "ParameterKey": "WorkerInstanceProfileName", 13
    "ParameterValue": "" 14
  },
  {
    "ParameterKey": "WorkerInstanceType", 15
    "ParameterValue": "m4.large" 16
  }
]
```

1 您的 Ignition 配置文件中为集群编码的集群基础架构名称。

2 指定从 Ignition 配置文件元数据中提取的基础架构名称，其格式为 **<cluster-name>-<random-string>**。

3 用于 worker 节点的当前 Red Hat Enterprise Linux CoreOS (RHCOS) AMI。

4 指定 **AWS::EC2::Image::Id** 值。

5 在其中启动 worker 节点的子网，最好是专用子网。

6 从 DNS 和负载均衡的 CloudFormation 模板输出的 **PrivateSubnets** 值指定子网。

7 与 worker 节点关联的 worker 安全组 ID。

8 指定安全组和角色的 CloudFormation 模板输出的 **WorkerSecurityGroupId** 值。

9 从中获取 bootstrap Ignition 配置文件的位置。

10 指定生成的 Ignition 配置的位置，**https://api-int.<cluster_name>.<domain_name>:22623/config/worker**。

11 要使用的 Base64 编码证书颁发机构字符串。

12 指定安装目录下 **worker.ign** 文件中的值。这个值是一个长字符串，格式为 **data:text/plain;charset=utf-8;base64,ABC...xYz==**。

13 与 worker 节点关联的 IAM 配置集。

14 指定安全组和角色的 CloudFormation 模板输出的 **WorkerInstanceProfile** 参数值。

15 用于 control plane 机器的 AWS 实例类型。

16 允许的值：

- **m4.large**
- **m4.xlarge**
- **m4.2xlarge**
- **m4.4xlarge**
- **m4.8xlarge**
- **m4.10xlarge**
- **m4.16xlarge**
- **m5.large**
- **m5.xlarge**
- **m5.2xlarge**
- **m5.4xlarge**
- **m5.8xlarge**
- **m5.10xlarge**

- **m5.16xlarge**
- **m6i.xlarge**
- **c4.2xlarge**
- **c4.4xlarge**
- **c4.8xlarge**
- **r4.large**
- **r4.xlarge**
- **r4.2xlarge**
- **r4.4xlarge**
- **r4.8xlarge**
- **r4.16xlarge**

重要

如果您的区域中没有 **m4** 实例类型，例如 **eu-west-3**，请改为使用 **m5** 类型。

2. 复制 **worker 机器的 CloudFormation 模板**一节中的模板，并将它以 YAML 文件形式保存到计算机上。此模板描述了集群所需的网络对象和负载均衡器。
3. 如果您将 **m5** 实例类型指定为 **WorkerInstanceType** 的值，请将该实例类型添加到 CloudFormation 模板中的 **WorkerInstanceType.AllowedValues** 参数。
4. 启动 CloudFormation 模板，以创建代表 worker 节点的 AWS 资源堆栈：

重要

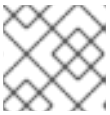
您必须在一行内输入命令。

```
$ aws cloudformation create-stack --stack-name <name> 1
    --template-body file://<template>.yaml \ 2
    --parameters file://<parameters>.json 3
```

1. **<name>** 是 CloudFormation 堆栈的名称，如 **cluster-worker-1**。如果您删除集群，则需要此堆栈的名称。
2. **<template>** 是您保存的 CloudFormation 模板 YAML 文件的相对路径和名称。
3. **<parameters>** 是 CloudFormation 参数 JSON 文件的相对路径和名称。

输出示例

```
arn:aws:cloudformation:us-east-1:269333783861:stack/cluster-worker-1/729ee301-1c2a-
11eb-348f-sd9888c65b59
```



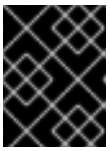
注意

CloudFormation 模板会创建一个堆栈，它代表一个 worker 节点。

5. 确认模板组件已存在：

```
$ aws cloudformation describe-stacks --stack-name <name>
```

6. 继续创建 worker 堆栈，直到为集群创建了充足的 worker 机器。您可以通过引用同一模板和参数文件并指定不同的堆栈名称来创建额外的 worker 堆栈。



重要

您必须至少创建两台 worker 机器，因此您必须创建至少两个使用此 CloudFormation 模板的堆栈。

2.10.14.1. worker 机器的 CloudFormation 模板

您可以使用以下 CloudFormation 模板来部署 OpenShift Container Platform 集群所需的 worker 机器。

例 2.30. worker 机器的 CloudFormation 模板

```
AWSTemplateFormatVersion: 2010-09-09
Description: Template for OpenShift Cluster Node Launch (EC2 worker instance)

Parameters:
  InfrastructureName:
    AllowedPattern: ^([a-zA-Z][a-zA-Z0-9\-]{0,26})$
    MaxLength: 27
    MinLength: 1
    ConstraintDescription: Infrastructure name must be alphanumeric, start with a letter, and have a
maximum of 27 characters.
    Description: A short, unique cluster ID used to tag nodes for the kubelet cloud provider.
    Type: String
  RhcosAmi:
    Description: Current Red Hat Enterprise Linux CoreOS AMI to use for bootstrap.
    Type: AWS::EC2::Image::Id
  Subnet:
    Description: The subnets, recommend private, to launch the master nodes into.
    Type: AWS::EC2::Subnet::Id
  WorkerSecurityGroupId:
    Description: The master security group ID to associate with master nodes.
    Type: AWS::EC2::SecurityGroup::Id
  IgnitionLocation:
    Default: https://api-int.$CLUSTER_NAME.$DOMAIN:22623/config/worker
    Description: Ignition config file location.
    Type: String
  CertificateAuthorities:
    Default: data:text/plain;charset=utf-8;base64,ABC...xYz==
    Description: Base64 encoded certificate authority string to use.
```

```
    Type: String
  WorkerInstanceProfileName:
    Description: IAM profile to associate with master nodes.
    Type: String
  WorkerInstanceType:
    Default: m5.large
    Type: String
    AllowedValues:
    - "m4.large"
    - "m4.xlarge"
    - "m4.2xlarge"
    - "m4.4xlarge"
    - "m4.10xlarge"
    - "m4.16xlarge"
    - "m5.large"
    - "m5.xlarge"
    - "m5.2xlarge"
    - "m5.4xlarge"
    - "m5.8xlarge"
    - "m5.12xlarge"
    - "m5.16xlarge"
    - "m5a.large"
    - "m5a.xlarge"
    - "m5a.2xlarge"
    - "m5a.4xlarge"
    - "m5a.8xlarge"
    - "m5a.10xlarge"
    - "m5a.16xlarge"
    - "c4.large"
    - "c4.xlarge"
    - "c4.2xlarge"
    - "c4.4xlarge"
    - "c4.8xlarge"
    - "c5.large"
    - "c5.xlarge"
    - "c5.2xlarge"
    - "c5.4xlarge"
    - "c5.9xlarge"
    - "c5.12xlarge"
    - "c5.18xlarge"
    - "c5.24xlarge"
    - "c5a.large"
    - "c5a.xlarge"
    - "c5a.2xlarge"
    - "c5a.4xlarge"
    - "c5a.8xlarge"
    - "c5a.12xlarge"
    - "c5a.16xlarge"
    - "c5a.24xlarge"
    - "r4.large"
    - "r4.xlarge"
    - "r4.2xlarge"
    - "r4.4xlarge"
    - "r4.8xlarge"
    - "r4.16xlarge"
    - "r5.large"
```

```
      - "r5.xlarge"
      - "r5.2xlarge"
      - "r5.4xlarge"
      - "r5.8xlarge"
      - "r5.12xlarge"
      - "r5.16xlarge"
      - "r5.24xlarge"
      - "r5a.large"
      - "r5a.xlarge"
      - "r5a.2xlarge"
      - "r5a.4xlarge"
      - "r5a.8xlarge"
      - "r5a.12xlarge"
      - "r5a.16xlarge"
      - "r5a.24xlarge"
      - "t3.large"
      - "t3.xlarge"
      - "t3.2xlarge"
      - "t3a.large"
      - "t3a.xlarge"
      - "t3a.2xlarge"

    Metadata:
      AWS::CloudFormation::Interface:
        ParameterGroups:
        - Label:
            default: "Cluster Information"
          Parameters:
          - InfrastructureName
        - Label:
            default: "Host Information"
          Parameters:
          - WorkerInstanceType
          - RhcosAmi
          - IgnitionLocation
          - CertificateAuthorities
          - WorkerSecurityGroupId
          - WorkerInstanceProfileName
        - Label:
            default: "Network Configuration"
          Parameters:
          - Subnet
        ParameterLabels:
          Subnet:
            default: "Subnet"
          InfrastructureName:
            default: "Infrastructure Name"
          WorkerInstanceType:
            default: "Worker Instance Type"
          WorkerInstanceProfileName:
            default: "Worker Instance Profile Name"
          RhcosAmi:
            default: "Red Hat Enterprise Linux CoreOS AMI ID"
          IgnitionLocation:
            default: "Worker Ignition Source"
          CertificateAuthorities:
```

```
        default: "Ignition CA String"
      WorkerSecurityGroupId:
        default: "Worker Security Group ID"

Resources:
  Worker0:
    Type: AWS::EC2::Instance
    Properties:
      ImageId: !Ref RhcosAmi
      BlockDeviceMappings:
      - DeviceName: /dev/xvda
        Ebs:
          VolumeSize: "120"
          VolumeType: "gp2"
      IamInstanceProfile: !Ref WorkerInstanceProfileName
      InstanceType: !Ref WorkerInstanceType
      NetworkInterfaces:
      - AssociatePublicIpAddress: "false"
        DeviceIndex: "0"
        GroupSet:
        - !Ref "WorkerSecurityGroupId"
        SubnetId: !Ref "Subnet"
      UserData:
        Fn::Base64: !Sub
        - '{"ignition":{"config":{"merge":[{"source":"${SOURCE}"}]},"security":{"tls":
{"certificateAuthorities":[{"source":"${CA_BUNDLE}"}]}},"version":"3.1.0"}}'
        - {
          SOURCE: !Ref IgnitionLocation,
          CA_BUNDLE: !Ref CertificateAuthorities,
        }
      Tags:
      - Key: !Join ["", ["kubernetes.io/cluster/", !Ref InfrastructureName]]
        Value: "shared"

Outputs:
  PrivateIP:
    Description: The compute node private IP address.
    Value: !GetAtt Worker0.PrivateIp
```

其他资源

- 您可以通过导航 AWS CloudFormation 控制台来查看您创建的 CloudFormation 堆栈的详情。

2.10.15. 使用用户置备的基础架构在 AWS 上初始化 bootstrap 序列

在 Amazon Web Services（AWS）中创建所有所需的基础架构后，您可以启动初始化 OpenShift Container Platform control plane 的 bootstrap 序列。

先决条件

- 已配置了一个 AWS 帐户。
- 您可以通过运行 **aws configure**，将 AWS 密钥和区域添加到本地 AWS 配置集中。

- 已为集群生成 Ignition 配置文件。
- 您在 AWS 中创建并配置了 VPC 及相关子网。
- 您在 AWS 中创建并配置了 DNS、负载均衡器和监听程序。
- 您在 AWS 中创建了集群所需的安全组和角色。
- 已创建 bootstrap 机器。
- 已创建 control plane 机器。
- 已创建 worker 节点。

流程

1. 更改为包含安装程序的目录，并启动初始化 OpenShift Container Platform control plane 的 bootstrap 过程：

```
$ ./openshift-install wait-for bootstrap-complete --dir <installation_directory> \ 1
    --log-level=info 2
```

1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

2 要查看不同的安装详情，请指定 **warn**、**debug** 或 **error**，而不要指定 **info**。

输出示例

```
INFO Waiting up to 20m0s for the Kubernetes API at
https://api.mycluster.example.com:6443...
INFO API v1.19.0+9f84db3 up
INFO Waiting up to 30m0s for bootstrapping to complete...
INFO It is now safe to remove the bootstrap resources
INFO Time elapsed: 1s
```

如果命令退出时没有 **FATAL** 警告，则 OpenShift Container Platform control plane 已被初始化。

注意

在 control plane 初始化后，它会设置计算节点，并以 Operator 的形式安装其他服务。

其他资源

- 如需了解在 OpenShift Container Platform 安装过程中监控安装、bootstrap 和 control plane 日志的详细信息，请参阅监控安装进度。
- 如需有关对 bootstrap 过程进行故障排除的信息，请参阅收集 bootstrap 节点诊断数据。
- 您可以使用 AWS EC2 控制台查看正在运行的实例的详情。

2.10.16. 通过下载二进制文件安装 OpenShift CLI

您需要安装 CLI（**oc**） 来使用命令行界面与 OpenShift Container Platform 进行交互。您可在 Linux 、Windows 或 macOS 上安装 **oc**。

重要

如果安装了旧版本的 **oc**，则无法使用 OpenShift Container Platform 4.6 中的所有命令。下载并安装新版本的 **oc**。

2.10.16.1. 在 Linux 上安装 OpenShift CLI

您可以按照以下流程在 Linux 上安装 OpenShift CLI（**oc**）二进制文件。

流程

1. 进入到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 在 **Version** 下拉菜单中选择相应的版本。
3. 单击 **OpenShift v4.6 Linux 客户端** 条目旁边的 **Download Now**，再保存文件。
4. 解包存档：

```
$ tar xvzf <file>
```

5. 把 **oc** 二进制代码放到 **PATH** 中的目录中。
   执行以下命令可以查看当前的 **PATH** 设置：

```
$ echo $PATH
```

安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
$ oc <command>
```

2.10.16.2. 在 Windows 上安装 OpenShift CLI

您可以按照以下流程在 Windows 上安装 OpenShift CLI（**oc**）二进制代码。

流程

1. 进入到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 在 **Version** 下拉菜单中选择相应的版本。
3. 单击 **OpenShift v4.6 Windows 客户端**条目旁边的 **Download Now**，再保存文件。
4. 使用 ZIP 程序解压存档。
5. 把 **oc** 二进制代码放到 **PATH** 中的目录中。
   要查看您的 **PATH**，请打开命令提示窗口并执行以下命令：

```
C:\> path
```

安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
C:\> oc <command>
```

2.10.16.3. 在 macOS 上安装 OpenShift CLI

您可以按照以下流程在 macOS 上安装 OpenShift CLI（**oc**）二进制代码。

流程

1. 进入到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 在 **Version** 下拉菜单中选择相应的版本。
3. 单击 **OpenShift v4.6 MacOSX 客户端**条目旁边的 **Download Now**，再保存文件。
4. 解包和解压存档。
5. 将 **oc** 二进制文件移到 PATH 的目录中。
   要查看您的 **PATH**，打开一个终端窗口并执行以下命令：

   ```
   $ echo $PATH
   ```

安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
$ oc <command>
```

2.10.17. 使用 CLI 登录到集群

您可以通过导出集群 **kubeconfig** 文件，以默认系统用户身份登录集群。**kubeconfig** 文件包含关于集群的信息，供 CLI 用于将客户端连接到正确集群和 API 服务器。该文件特只适用于一个特定的集群，在 OpenShift Container Platform 安装过程中创建。

先决条件

- 已部署了 OpenShift Container Platform 集群。
- 已安装 **oc** CLI。

流程

1. 导出 **kubeadmin** 凭证：

   ```
   $ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
   ```

   1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

2. 使用导出的配置，验证能否成功运行 **oc** 命令：

   ```
   $ oc whoami
   ```

   输出示例

```
system:admin
```

2.10.18. 批准机器的证书签名请求

将机器添加到集群时，会为您添加的每台机器生成两个待处理证书签名请求（CSR）。您必须确认这些 CSR 已获得批准，或根据需要自行批准。客户端请求必须首先被批准，然后是服务器请求。

先决条件

- 您已将机器添加到集群中。

流程

1. 确认集群可以识别这些机器：

```
$ oc get nodes
```

输出示例

```
NAME      STATUS    ROLES   AGE  VERSION
master-0  Ready     master  63m  v1.19.0
master-1  Ready     master  63m  v1.19.0
master-2  Ready     master  64m  v1.19.0
```

输出将列出您创建的所有机器。

注意

在一些 CSR 被批准前，以上输出可能不包括计算节点（也称为 worker 节点）。

2. 检查待处理的 CSR，并确保可以看到添加到集群中的每台机器都有 **Pending** 或 **Approved** 状态的客户端请求：

```
$ oc get csr
```

输出示例

```
NAME        AGE     REQUESTOR                                                   CONDITION
csr-8b2br   15m     system:serviceaccount:openshift-machine-config-operator:node-
bootstrapper   Pending
csr-8vnps   15m     system:serviceaccount:openshift-machine-config-operator:node-
bootstrapper   Pending
...
```

在本例中，两台机器加入了集群。您可能会在列表中看到更多已批准的 CSR。

3. 如果 CSR 没有获得批准，请在所添加机器的所有待处理 CSR 都处于 **Pending** 状态后，为您的集群机器批准这些 CSR：

注意

由于 CSR 会自动轮转，因此请在将机器添加到集群后一小时内批准您的 CSR。如果没有在一小时内批准，证书将会轮转，每个节点将会存在多个证书。您必须批准所有这些证书。批准客户端 CSR 后， Kubelet 为服务证书创建辅助 CSR，这需要手动批准。然后，如果 Kubelet 请求具有相同参数的新证书，则 **machine-approver** 会自动批准后续服务证书续订请求。

注意

对于在未启用机器 API 的平台中运行的集群，如裸机和其他用户置备的基础架构，必须采用一种方法自动批准 kubelet 提供证书请求（CSR）。如果没有批准请求，则 **oc exec**、**oc rsh** 和 **oc logs** 命令将无法成功，因为 API 服务器连接到 kubelet 时需要服务证书。与 Kubelet 端点联系的任何操作都需要此证书批准。这个方法必须监视新的 CSR，确认 CSR 由 **system:node** 或 **system:admin** 组中的 **node-bootstrapper** 服务帐户提交，并确认节点的身份。

- 若要单独批准，请对每个有效的 CSR 运行以下命令：

```
$ oc adm certificate approve <csr_name> 1
```

1 **<csr_name>** 是当前 CSR 列表中 CSR 的名称。

- 要批准所有待处理的 CSR，请运行以下命令：

```
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}{{"\n"}}
{{end}}{{end}}' | xargs --no-run-if-empty oc adm certificate approve
```

注意

在有些 CSR 被批准前，一些 Operator 可能无法使用。

4. 现在，您的客户端请求已被批准，您必须查看添加到集群中的每台机器的服务器请求：

```
$ oc get csr
```

输出示例

```
NAME        AGE    REQUESTOR                                                   CONDITION
csr-bfd72   5m26s  system:node:ip-10-0-50-126.us-east-2.compute.internal
Pending
csr-c57lv   5m26s  system:node:ip-10-0-95-157.us-east-2.compute.internal
Pending
...
```

5. 如果剩余的 CSR 没有被批准，且处于 **Pending** 状态，请批准集群机器的 CSR：

- 若要单独批准，请对每个有效的 CSR 运行以下命令：

```
$ oc adm certificate approve <csr_name> 1
```

1 **<csr_name>** 是当前 CSR 列表中 CSR 的名称。

- 要批准所有待处理的 CSR，请运行以下命令：

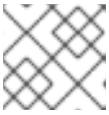
```
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}{{"\n"}}
{{end}}{{end}}' | xargs oc adm certificate approve
```

6. 批准所有客户端和服务器 CSR 后，器将处于 **Ready** 状态。运行以下命令验证：

```
$ oc get nodes
```

输出示例

```
NAME      STATUS    ROLES   AGE  VERSION
master-0  Ready     master  73m  v1.20.0
master-1  Ready     master  73m  v1.20.0
master-2  Ready     master  74m  v1.20.0
worker-0  Ready     worker  11m  v1.20.0
worker-1  Ready     worker  11m  v1.20.0
```



注意

批准服务器 CSR 后可能需要几分钟时间让机器转换为 **Ready** 状态。

其他信息

- 如需有关 CSR 的更多信息，请参阅证书签名请求。

2.10.19. 初始 Operator 配置

在 control plane 初始化后，您必须立即配置一些 Operator 以便它们都可用。

先决条件

- 您的 control plane 已初始化。

流程

1. 观察集群组件上线：

```
$ watch -n5 oc get clusteroperators
```

输出示例

```
NAME                                       VERSION AVAILABLE   PROGRESSING   DEGRADED
SINCE
authentication                              4.6.0   True        False         False      3h56m
cloud-credential                            4.6.0   True        False         False      29h
cluster-autoscaler                          4.6.0   True        False         False      29h
config-operator                             4.6.0   True        False         False      6h39m
console                                    4.6.0   True        False         False      3h59m
csi-snapshot-controller                          4.6.0   True        False         False      4h12m
```

```
dns                                        4.6.0   True        False         False      4h15m
etcd                                       4.6.0   True        False         False      29h
image-registry                             4.6.0   True        False         False      3h59m
ingress                                    4.6.0   True        False         False      4h30m
insights                                   4.6.0   True        False         False      29h
kube-apiserver                             4.6.0   True        False         False      29h
kube-controller-manager                    4.6.0   True        False         False      29h
kube-scheduler                             4.6.0   True        False         False      29h
kube-storage-version-migrator              4.6.0   True        False         False      4h2m
machine-api                                4.6.0   True        False         False      29h
machine-approver                           4.6.0   True        False         False      6h34m
machine-config                             4.6.0   True        False         False      3h56m
marketplace                                4.6.0   True        False         False      4h2m
monitoring                                 4.6.0   True        False         False      6h31m
network                                    4.6.0   True        False         False      29h
node-tuning                                4.6.0   True        False         False      4h30m
openshift-apiserver                        4.6.0   True        False         False      3h56m
openshift-controller-manager               4.6.0   True        False         False      4h36m
openshift-samples                          4.6.0   True        False         False      4h30m
operator-lifecycle-manager                 4.6.0   True        False         False      29h
operator-lifecycle-manager-catalog         4.6.0   True        False         False      29h
operator-lifecycle-manager-packageserver   4.6.0   True        False         False      3h59m
service-ca                                 4.6.0   True        False         False      29h
storage                                    4.6.0   True        False         False      4h30m
```

2. 配置不可用的 Operator。

2.10.19.1. 镜像 registry 存储配置

Amazon Web Services 提供默认存储，这意味着 Image Registry Operator 在安装后可用。但是，如果 Registry Operator 无法创建 S3 存储桶并自动配置存储，您需要手工配置 registry 存储。

示配置生产集群所需的持久性卷的说明。如果适用，显示有关将空目录配置为存储位置的说明，该位置只可用于非生产集群。

另外还提供了在升级过程中使用 **Recreate** rollout 策略来允许镜像 registry 使用块存储类型的说明。

您可以在 AWS 中为用户置备的基础架构配置 registry 存储,以将 OpenShift Container Platform 部署到隐藏的区域。请参阅为 AWS 用户置备的基础架构配置 registry。

2.10.19.1.1. 为使用用户置备的基础架构的 AWS 配置 registry 存储

在安装过程中，使用您的云凭据就可以创建一个 Amazon S3 存储桶，Registry Operator 将会自动配置存储。

如果 Registry Operator 无法创建 S3 存储桶或自动配置存储，您可以按照以下流程创建 S3 存储桶并配置存储。

先决条件

- 在带有用户置备的基础架构的 AWS 上有一个集群。
- 对于 Amazon S3 存储，secret 应该包含以下两个键：
  - **REGISTRY_STORAGE_S3_ACCESSKEY**

- **REGISTRY_STORAGE_S3_SECRETKEY**

流程

如果 Registry Operator 无法创建 S3 存储桶并自动配置存储，请进行以下操作。

1. 设置一个 Bucket Lifecycle Policy 用来终止已有一天之久的未完成的分段上传操作。
2. 在**configs.imageregistry.operator.openshift.io/cluster**中中输入存储配置：

```
$ oc edit configs.imageregistry.operator.openshift.io/cluster
```

配置示例

```
storage:
  s3:
    bucket: <bucket-name>
    region: <region-name>
```

警告

为了保护 AWS 中 registry 镜像的安全，阻止对 S3 存储桶的公共访问。

2.10.19.1.2. 在非生产集群中配置镜像 registry 存储

您必须为 Image Registry Operator 配置存储。对于非生产集群，您可以将镜像 registry 设置为空目录。如果您这样做，重启 registry 后会丢失所有镜像。

流程

- 将镜像 registry 存储设置为空目录：

```
$ oc patch configs.imageregistry.operator.openshift.io cluster --type merge --patch '{"spec":{"storage":{"emptyDir":{}}}}'
```

警告

仅可为非生产集群配置这个选项。

如果在 Image Registry Operator 初始化其组件前运行此命令，**oc patch** 命令会失败并显示以下错误：

```
Error from server (NotFound): configs.imageregistry.operator.openshift.io "cluster" not found
```

等待几分钟，然后再次运行该命令。

2.10.20. 删除 bootstrap 资源：

完成集群的初始 Operator 配置后，从 Amazon Web Services (AWS) 中删除 bootstrap 资源。

先决条件

- 已为集群完成初始的 Operator 配置。

流程

1. 删除 bootstrap 资源。如果您使用了 CloudFormation 模板，请删除其堆栈：
   - 使用 AWS CLI 删除堆栈：

     ```
     $ aws cloudformation delete-stack --stack-name <name> 1
     ```

     1 **<name>** 是 bootstrap 堆栈的名称。

   - 使用 AWS CloudFormation 控制台删除堆栈。

2.10.21. 创建 Ingress DNS 记录

如果您删除了 DNS 区配置，请手动创建指向 Ingress 负载均衡器的 DNS 记录。您可以创建一个 wildcard 记录或具体的记录。以下流程使用了 A 记录，但您可以使用其他所需记录类型，如 CNAME 或别名。

先决条件

- 已在 Amazon Web Services (AWS) 上安装了使用您置备的基础架构的 OpenShift Container Platform 集群。
- 已安装 OpenShift CLI（**oc**）。
- 安装了 **jq** 软件包。
- 您下载了 AWS CLI 并安装到您的计算机上。请参阅使用捆绑安装程序（Linux、macOS 或 Unix）安装 AWS CLI的文档。

流程

1. 决定要创建的路由。
   - 要创建一个 wildcard 记录，请使用 ***.apps.<cluster_name>.<domain_name>**，其中 **<cluster_name>** 是集群名称，**<domain_name>** 是 OpenShift Container Platform 集群的 Route 53 基域。
   - 要创建特定的记录，您必须为集群使用的每个路由创建一个记录，如下所示：

     ```
     $ oc get --all-namespaces -o jsonpath='{range .items[*]}{range .status.ingress[*]}{.host}
     {"\n"}{end}{end}' routes
     ```

     输出示例

     ```
     -
     ```

```
oauth-openshift.apps.<cluster_name>.<domain_name>
console-openshift-console.apps.<cluster_name>.<domain_name>
downloads-openshift-console.apps.<cluster_name>.<domain_name>
alertmanager-main-openshift-monitoring.apps.<cluster_name>.<domain_name>
grafana-openshift-monitoring.apps.<cluster_name>.<domain_name>
prometheus-k8s-openshift-monitoring.apps.<cluster_name>.<domain_name>
```

2. 获取 Ingress Operator 负载均衡器状态，并记录其使用的外部 IP 地址值，如 **EXTERNAL-IP** 列所示：

```
$ oc -n openshift-ingress get service router-default
```

输出示例

```
NAME            TYPE           CLUSTER-IP     EXTERNAL-IP                         PORT(S)
AGE
router-default  LoadBalancer   172.30.62.215  ab3...28.us-east-2.elb.amazonaws.com
80:31499/TCP,443:30693/TCP   5m
```

3. 为负载均衡器定位托管区 ID：

```
$ aws elb describe-load-balancers | jq -r '.LoadBalancerDescriptions[] | select(.DNSName ==
"<external_ip>").CanonicalHostedZoneNameID' 1
```

1 对于 **<external_ip>**，请指定您获取的 Ingress Operator 负载均衡器的外部 IP 地址值。

输出示例

```
Z3AADJGX6KTTL2
```

这个命令的输出是负载均衡器托管区 ID。

4. 获取集群域的公共托管区 ID：

```
$ aws route53 list-hosted-zones-by-name \
        --dns-name "<domain_name>" \ 1
        --query 'HostedZones[? Config.PrivateZone != `true` && Name ==
`<domain_name>.`].Id' 2
        --output text
```

1 2 对于 **<domain_name>**，请为 OpenShift Container Platform 集群指定 Route 53 基域。

输出示例

```
/hostedzone/Z3URY6TWQ91KVV
```

命令输出中会显示您的域的公共托管区 ID。在本例中是 **Z3URY6TWQ91KVV**。

5. 在您的私有区中添加别名记录：

```
$ aws route53 change-resource-record-sets --hosted-zone-id "<private_hosted_zone_id>" --
```

```
change-batch '{ (1)
>   "Changes": [
>     {
>       "Action": "CREATE",
>       "ResourceRecordSet": {
>         "Name": "\\052.apps.<cluster_domain>", (2)
>         "Type": "A",
>         "AliasTarget":{
>           "HostedZoneId": "<hosted_zone_id>", (3)
>           "DNSName": "<external_ip>.", (4)
>           "EvaluateTargetHealth": false
>         }
>       }
>     }
>   ]
> }'
```

(1) 对于 **<private_hosted_zone_id>**，指定 DNS 和负载均衡的 CloudFormation 模板输出的值。

(2) 对于 **<cluster_domain>**，请指定用于 OpenShift Container Platform 集群的域或子域。

(3) 对于 **<hosted_zone_id>**，请为您获得的负载均衡器指定公共托管区 ID。

(4) 对于 **<external_ip>**，请指定 Ingress Operator 负载均衡器的外部 IP 地址值。请确定在该参数值中包含最后的句点（.）。

6. 在您的公共区中添加记录：

```
$ aws route53 change-resource-record-sets --hosted-zone-id "<public_hosted_zone_id>"" --
change-batch '{ (1)
>   "Changes": [
>     {
>       "Action": "CREATE",
>       "ResourceRecordSet": {
>         "Name": "\\052.apps.<cluster_domain>", (2)
>         "Type": "A",
>         "AliasTarget":{
>           "HostedZoneId": "<hosted_zone_id>", (3)
>           "DNSName": "<external_ip>.", (4)
>           "EvaluateTargetHealth": false
>         }
>       }
>     }
>   ]
> }'
```

(1) 对于 **<public_hosted_zone_id>**，请为您的域指定公共托管区。

(2) 对于 **<cluster_domain>**，请指定用于 OpenShift Container Platform 集群的域或子域。

(3) 对于 **<hosted_zone_id>**，请为您获得的负载均衡器指定公共托管区 ID。

(4) 对于 **<external_ip>**，请指定 Ingress Operator 负载均衡器的外部 IP 地址值。请确定在该参数值中包含最后的句点（.）。

2.10.22. 在用户置备的基础架构上完成 AWS 安装

在用户置备的基础架构 Amazon Web Service (AWS) 上启动 OpenShift Container Platform 安装后，监视进程并等待安装完成。

先决条件

- 您在用户置备的 AWS 基础架构上为 OpenShift Container Platform 集群删除了 bootstrap 节点。
- 已安装 **oc** CLI。

流程

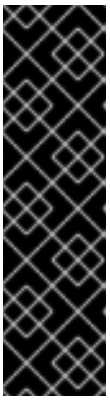
- 在包含安装程序的目录中完成集群安装：

```
$ ./openshift-install --dir <installation_directory> wait-for install-complete 1
```

1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

输出示例

```
INFO Waiting up to 40m0s for the cluster at https://api.mycluster.example.com:6443 to
initialize...
INFO Waiting up to 10m0s for the openshift-console route to be created...
INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run 'export
KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'
INFO Access the OpenShift web-console here: https://console-openshift-
console.apps.mycluster.example.com
INFO Login to the console with user: "kubeadmin", and password: "4vYBz-Fe5en-ymBEc-
Wt6NL"
INFO Time elapsed: 1s
```



重要

- 安装程序生成的 Ignition 配置文件包含在 24 小时后过期的证书，然后在过期时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外情况是，您需要手动批准待处理的 **node-bootstrapper** 证书签名请求（CSR）来恢复 kubelet 证书。如需更多信息，请参阅*从过期的 control plane 证书中恢复*的文档。
- 建议您在生成 12 小时后使用 Ignition 配置文件，因为集群安装后 24 小时证书从 16 小时轮转至 22 小时。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中运行证书更新时避免安装失败。

2.10.23. 使用 Web 控制台登录到集群

kubeadmin 用户默认在 OpenShift Container Platform 安装后存在。您可以使用 OpenShift Container Platform Web 控制台以 **kubeadmin** 用户身份登录集群。

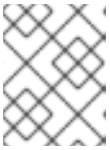
先决条件

- 有访问安装主机的访问权限。
- 您完成了集群安装，所有集群 Operator 都可用。

流程

1. 从安装主机上的 **kubeadmin -password** 文件中获取 kubeadmin 用户的密码：

```
$ cat <installation_directory>/auth/kubeadmin-password
```



注意

另外，您还可以从安装主机上的 **<installation_directory>/.openshift_install.log** 日志文件获取 **kubeadmin** 密码。

2. 列出 OpenShift Container Platform Web 控制台路由：

```
$ oc get routes -n openshift-console | grep 'console-openshift'
```

注意

另外，您还可以从安装主机上的 **<installation_directory>/.openshift_install.log 日志** 文件获取 OpenShift Container Platform 路由。

输出示例

```
console     console-openshift-console.apps.<cluster_name>.<base_domain>           console
https   reencrypt/Redirect   None
```

3. 在 Web 浏览器中导航到上一命令输出中包括的路由，以 **kubeadmin** 用户身份登录。

其他资源

- 如需有关访问和了解 OpenShift Container Platform Web 控制台的更多信息，请参阅访问 Web 控制台。

2.10.24. OpenShift Container Platform 的 Telemetry 访问

在 OpenShift Container Platform 4.6 中，默认运行的 Telemetry 服务提供有关集群健康状况和成功更新的指标，需要访问互联网。如果您的集群连接到互联网，Telemetry 会自动运行，而且集群会注册到 OpenShift Cluster Manager。

确认 OpenShift Cluster Manager 清单正确后，可以由 Telemetry 自动维护，也可以使用 OpenShift Cluster Manager 手动维护，使用订阅监控来跟踪帐户或多集群级别的 OpenShift Container Platform 订阅。

其他资源

- 有关 Telemetry 服务的更多信息，请参阅关于远程健康监控。

2.10.25. 其他资源

- 如需有关 AWS CloudFormation 堆栈的更多信息，请参阅 AWS 文档中的使用堆栈。

2.10.26. 后续步骤

- 验证安装。
- 自定义集群。
- 如果需要，您可以选择不使用远程健康报告。
- 如果需要，您可以删除云供应商凭证。

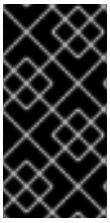
2.11. 在带有用户置备的受限网络中的 AWS 上安装集群

在 OpenShift Container Platform 版本 4.6 中，您可以使用您提供的基础架构和安装发行内容的内部镜像在 Amazon Web Services（AWS）上安装集群。

重要

虽然您可以使用镜像安装发行内容安装 OpenShift Container Platform 集群，但您的集群仍需要访问互联网才能使用 AWS API。

创建此基础架构的一种方法是使用提供的 CloudFormation 模板。您可以修改模板来自定义基础架构，或使用其包含的信息来按照公司策略创建 AWS 对象。



重要

进行用户置备的基础架构安装的步骤仅作为示例。使用您提供的基础架构安装集群需要了解云供应商和 OpenShift Container Platform 安装过程。提供的几个 CloudFormation 模板可帮助完成这些步骤，或者帮助您自行建模。您也可以自由选择通过其他方法创建所需的资源；模板仅作示例之用。

2.11.1. 先决条件

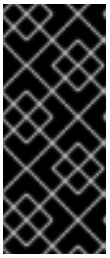
- 您在镜像主机上创建了一个镜像 registry，并获取您的 OpenShift Container Platform 版本的 **imageContentSources** 数据。

  重要

  由于安装介质位于堡垒主机上，因此请使用该计算机完成所有安装步骤。

- 您可以参阅有关 OpenShift Container Platform 安装和更新流程 的详细信息。
- 已将 AWS 帐户配置为托管集群。



重要

如果您的计算机上存储有 AWS 配置集，则不要在使用多因素验证设备的同时使用您生成的临时会话令牌。在集群的整个生命周期中，集群会持续使用您的当前 AWS 凭证来创建 AWS 资源，因此您必须使用基于密钥的长期凭证。要生成适当的密钥，请参阅 AWS 文档中的管理 IAM 用户的访问密钥。您可在运行安装程序时提供密钥。

- 您下载了 AWS CLI 并安装到您的计算机上。请参阅 AWS 文档中的使用捆绑安装程序（Linux、macOS 或 Unix）安装 AWS CLI。
- 如果使用防火墙并计划使用 Telemetry 服务，需要 将防火墙配置为允许集群需要访问的站点。

注意

如果您要配置代理，请务必也要查看此站点列表。

- 如果不允许系统管理身份和访问管理（IAM），集群管理员可以 手动创建和维护 IAM 凭证。手动模式也可以用于云 IAM API 无法访问的环境中。

2.11.2. 关于在受限网络中安装

在 OpenShift Container Platform 4.6 中，可以执行不需要有效的互联网连接来获取软件组件的安装。受限网络安装可使用安装程序置备的基础架构或用户置备的基础架构完成，具体取决于您要安装集群的云平台。

如果选择在云平台中执行受限网络安装，仍然需要访问其云 API。有些云功能，比如 Amazon Web Service 的 Route 53 DNS 和 IAM 服务，需要访问互联网。根据您的网络，在裸机硬件或 VMware vSphere 上安装时可能需要较少的互联网访问。

要完成受限网络安装，您必须创建一个 registry，镜像 OpenShift Container Platform registry 的内容并包含其安装介质。您可以在堡垒主机上创建此镜像，该主机可同时访问互联网和您的封闭网络，也可以使用满足您的限制条件的其他方法。

重要

由于用户置备安装配置的复杂性，在尝试使用用户置备的基础架构受限网络安装前，请考虑完成标准用户置备的基础架构安装。通过完成此测试安装，您可以更轻松地隔离和排查您在受限网络中安装时可能出现的问题。

2.11.2.1. 其他限制

受限网络中的集群还有以下额外限制：

- **ClusterVersion** 状态包含一个 **Unable to retrieve available updates** 错误。
- 默认情况下，您无法使用 Developer Catalog 的内容，因为您无法访问所需的镜像流标签。

2.11.3. OpenShift Container Platform 的互联网访问

在 OpenShift Container Platform 4.6 中，您需要访问互联网来获得用来安装集群的镜像。

您必须具有以下互联网访问权限：

- 访问 OpenShift Cluster Manager 以下载安装程序并执行订阅管理。如果集群可以访问互联网，并且没有禁用 Telemetry，该服务会自动授权您的集群。
- 访问 Quay.io，以获取安装集群所需的软件包。
- 获取执行集群更新所需的软件包。

重要

如果您的集群无法直接访问互联网，则可以在置备的某些类基础架构上执行受限网络安装。在此过程中，您要下载所需的内容，并使用它在镜像 registry（mirror registry） 中填充安装集群并生成安装程序所需的软件包。对于某些安装类型，集群要安装到的环境不需要访问互联网。在更新集群之前，要更新 registry 镜像系统中的内容。

2.11.4. 所需的 AWS 基础架构组件

要在 Amazon Web Services (AWS) 中用户置备的基础架构上安装 OpenShift Container Platform，您必须手动创建机器及其支持的基础架构。

如需有关不同平台集成测试的更多信息，请参阅 OpenShift Container Platform 4.x Tested Integrations 页面。

通过使用提供的 CloudFormation 模板，您可以创建代表以下组件的 AWS 资源堆栈：

- 一个 AWS Virtual Private Cloud (VPC)
- 网络和负载均衡组件
- 安全组和角色
- 一个 OpenShift Container Platform bootstrap 节点
- OpenShift Container Platform control plane 节点
- 一个 OpenShift Container Platform 计算节点

或者，您可以手动创建组件，也可以重复使用满足集群要求的现有基础架构。查看 CloudFormation 模板，了解组件如何相互连接的更多详情。

2.11.4.1. 集群机器

以下机器需要 **AWS::EC2::Instance** 对象：

- bootstrap 机器。安装过程中需要此机器，但可在集群部署后删除。
- 三个 control plane 机器。control plane 机器不受机器集的管控。
- 计算机器。在安装过程中创建至少两台计算（compute）机器（也称为 worker 机器）。这些机器不受机器集的管控。

您可以通过提供的 CloudFormation 模板，为集群机器使用以下实例类型。

重要

如果您的区域中没有 **m4** 实例类型，例如 **eu-west-3**，请改为使用 **m5** 类型。

表 2.32. 机器的实例类型

实例类型	bootstrap	Control plane	Compute
i3.large	x		
m4.large			x
m4.xlarge		x	x
m4.2xlarge		x	x
m4.4xlarge		x	x
m4.8xlarge		x	x
m4.10xlarge		x	x
m4.16xlarge		x	x
m5.large			x
m5.xlarge		x	x
m5.2xlarge		x	x
m5.4xlarge		x	x
m5.8xlarge		x	x
m5.10xlarge		x	x
m5.16xlarge		x	x
m6i.xlarge		x	x
c4.2xlarge		x	x
c4.4xlarge		x	x
c4.8xlarge		x	x
r4.large			x
r4.xlarge		x	x
r4.2xlarge		x	x

实例类型	bootstrap	Control plane	Compute
r4.4xlarge		x	x
r4.8xlarge		x	x
r4.16xlarge		x	x

您可能能够使用符合这些实例类型规格的其他实例类型。

2.11.4.2. 其他基础架构组件

- VPC
- DNS 条目
- 负载均衡器（典型或网络）和监听器
- 公共和专用路由 53 区域
- 安全组
- IAM 角色
- S3 存储桶

如果您在断开连接的环境或使用代理的环境中工作，则无法访问 EC2 和 ELB 端点的公共 IP 地址。要访问这些端点，您必须创建一个 VPC 端点，并将其附加到集群使用的子网。创建以下端点：

- **ec2.<region>.amazonaws.com**
- **elasticloadbalancing.<region>.amazonaws.com**
- **s3.<region>.amazonaws.com**

所需的 VPC 组件

您必须提供合适的 VPC 和子网，以便与您的机器通信。

组件	AWS 类型	描述
VPC	• **AWS::EC2::VPC** • **AWS::EC2::VPCEndpoint**	您必须提供一个公共 VPC 供集群使用。VPC 使用引用每个子网的路由表的端点，以改进与托管在 S3 中的 registry 的通信。
公共子网	• **AWS::EC2::Subnet** • **AWS::EC2::SubnetNetworkAclAssociation**	您的 VPC 必须有 1 到 3 个可用区的公共子网，并将其与适当的入口规则关联。

<table>
<tr><th>组件</th><th>AWS 类型</th><th colspan="2">描述</th></tr>
<tr><td>互联网网关</td><td>AWS::EC2::InternetGateway
AWS::EC2::VPCGatewayAttachment
AWS::EC2::RouteTable
AWS::EC2::Route
AWS::EC2::SubnetRouteTableAssociation
AWS::EC2::NatGateway
AWS::EC2::EIP</td><td colspan="2">您必须有一个公共互联网网关，以及附加到 VPC 的公共路由。在提供的模板中，每个公共子网都有一个具有 EIP 地址的 NAT 网关。这些 NAT 网关允许集群资源（如专用子网实例）访问互联网，而有些受限网络或代理场景则不需要它们。</td></tr>
<tr><td>网络访问控制</td><td>AWS::EC2::NetworkAcl
AWS::EC2::NetworkAclEntry</td><td colspan="2">您必须允许 VPC 访问下列端口：</td></tr>
<tr><td></td><td></td><td>端口</td><td>原因</td></tr>
<tr><td></td><td></td><td>80</td><td>入站 HTTP 流量</td></tr>
<tr><td></td><td></td><td>443</td><td>入站 HTTPS 流量</td></tr>
<tr><td></td><td></td><td>22</td><td>入站 SSH 流量</td></tr>
<tr><td></td><td></td><td>1024 - 65535</td><td>入站临时流量</td></tr>
<tr><td></td><td></td><td>0 - 65535</td><td>出站临时流量</td></tr>
<tr><td>专用子网</td><td>AWS::EC2::Subnet
AWS::EC2::RouteTable
AWS::EC2::SubnetRouteTableAssociation</td><td colspan="2">您的 VPC 可以具有私有子网。提供的 CloudFormation 模板可为 1 到 3 个可用区创建专用子网。如果您使用专用子网，必须为其提供适当的路由和表。</td></tr>
</table>

所需的 DNS 和负载均衡组件

您的 DNS 和负载均衡器配置需要使用公共托管区，并可使用类似安装程序使用的专用托管区（如果安装程序置备了集群的基础架构）。您必须创建一个解析到负载均衡器的 DNS 条目。**api.<cluster_name>.<domain>** 的条目必须指向外部负载均衡器，**api-int.<cluster_name>.<domain>** 的条目则必须指向内部负载均衡器。

集群还需要负载均衡器，以及监听端口 6443（用于 Kubernetes API 及其扩展）和端口 22623（用于新机器的 Ignition 配置文件）的监听程序。目标是 control plane 节点（也称为 master 节点）。集群外的客户端和集群内的节点都必须能够访问端口 6443。集群内的节点必须能够访问端口 22623。

组件	AWS 类型	描述
DNS	**AWS::Route53::HostedZone**	内部 DNS 的托管区。
etcd 记录集	**AWS::Route53::RecordSet**	control plane 机器的 etcd 注册记录。
公共负载均衡器	**AWS::ElasticLoadBalancingV2::LoadBalancer**	公共子网的负载均衡器。
外部 API 服务器记录	**AWS::Route53::RecordSetGroup**	外部 API 服务器的别名记录。
外部监听程序	**AWS::ElasticLoadBalancingV2::Listener**	为外部负载均衡器监听端口 6443 的监听程序。
外部目标组	**AWS::ElasticLoadBalancingV2::TargetGroup**	外部负载均衡器的目标组。
专用负载均衡器	**AWS::ElasticLoadBalancingV2::LoadBalancer**	专用子网的负载均衡器。
内部 API 服务器记录	**AWS::Route53::RecordSetGroup**	内部 API 服务器的别名记录。
内部监听程序	**AWS::ElasticLoadBalancingV2::Listener**	为内部负载均衡器监听端口 22623 的监听程序。
内部目标组	**AWS::ElasticLoadBalancingV2::TargetGroup**	内部负载均衡器的目标组。

组件	AWS 类型	描述
内部监听程序	**AWS::ElasticLoadBalancingV2::Listener**	为内部负载均衡器监听端口 6443 的监听程序。
内部目标组	**AWS::ElasticLoadBalancingV2::TargetGroup**	内部负载均衡器的目标组。

安全组

control plane 和 worker 机器需要访问下列端口：

组	类型	IP 协议	端口范围
MasterSecurityGroup	**AWS::EC2::SecurityGroup**	**icmp**	**0**
		tcp	**22**
		tcp	**6443**
		tcp	**22623**
WorkerSecurityGroup	**AWS::EC2::SecurityGroup**	**icmp**	**0**
		tcp	**22**
BootstrapSecurityGroup	**AWS::EC2::SecurityGroup**	**tcp**	**22**
		tcp	**19531**

control plane 入口

control plane 机器需要以下入口组。每个入口组都是 **AWS::EC2::SecurityGroupIngress** 资源。

入口组	描述	IP 协议	端口范围
MasterIngressEtcd	etcd	**tcp**	**2379- 2380**
MasterIngressVxlan	Vxlan 数据包	**udp**	**4789**
MasterIngressWorkerVxlan	Vxlan 数据包	**udp**	**4789**

入口组	描述	IP 协议	端口范围
MasterIngressInternal	内部集群通信和 Kubernetes 代理指标	**tcp**	**9000 - 9999**
MasterIngressWorkerInternal	内部集群通信	**tcp**	**9000 - 9999**
MasterIngressKube	kubernetes kubelet、调度程序和控制器管理器	**tcp**	**10250 - 10259**
MasterIngressWorkerKube	kubernetes kubelet、调度程序和控制器管理器	**tcp**	**10250 - 10259**
MasterIngressIngressServices	Kubernetes 入口服务	**tcp**	**30000 - 32767**
MasterIngressWorkerIngressServices	Kubernetes 入口服务	**tcp**	**30000 - 32767**
MasterIngressGeneve	Geneve 数据包	**udp**	**6081**
MasterIngressWorkerGeneve	Geneve 数据包	**udp**	**6081**
MasterIngressIpsecIke	IPsec IKE 数据包	**udp**	**500**
MasterIngressWorkerIpsecIke	IPsec IKE 数据包	**udp**	**500**
MasterIngressIpsecNat	IPsec NAT-T 数据包	**udp**	**4500**
MasterIngressWorkerIpsecNat	IPsec NAT-T 数据包	**udp**	**4500**
MasterIngressIpsecEsp	IPsec ESP 数据包	**50**	**All**
MasterIngressWorkerIpsecEsp	IPsec ESP 数据包	**50**	**All**

入口组	描述	IP 协议	端口范围
MasterIngressInternalUDP	内部集群通信	**udp**	**9000 - 9999**
MasterIngressWorkerInternalUDP	内部集群通信	**udp**	**9000 - 9999**
MasterIngressIngressServicesUDP	Kubernetes 入口服务	**udp**	**30000 - 32767**
MasterIngressWorkerIngressServicesUDP	Kubernetes 入口服务	**udp**	**30000 - 32767**

worker 入口

worker 机器需要以下入口组。每个入口组都是 **AWS::EC2::SecurityGroupIngress** 资源。

入口组	描述	IP 协议	端口范围
WorkerIngressVxlan	Vxlan 数据包	**udp**	**4789**
WorkerIngressWorkerVxlan	Vxlan 数据包	**udp**	**4789**
WorkerIngressInternal	内部集群通信	**tcp**	**9000 - 9999**
WorkerIngressWorkerInternal	内部集群通信	**tcp**	**9000 - 9999**
WorkerIngressKube	Kubernetes kubelet、调度程序和控制器管理器	**tcp**	**10250**
WorkerIngressWorkerKube	Kubernetes kubelet、调度程序和控制器管理器	**tcp**	**10250**
WorkerIngressIngressServices	Kubernetes 入口服务	**tcp**	**30000 - 32767**
WorkerIngressWorkerIngressServices	Kubernetes 入口服务	**tcp**	**30000 - 32767**

入口组	描述	IP 协议	端口范围
WorkerIngressGeneve	Geneve 数据包	**udp**	**6081**
WorkerIngressMasterGeneve	Geneve 数据包	**udp**	**6081**
WorkerIngressIpsecIke	IPsec IKE 数据包	**udp**	**500**
WorkerIngressMasterIpsecIke	IPsec IKE 数据包	**udp**	**500**
WorkerIngressIpsecNat	IPsec NAT-T 数据包	**udp**	**4500**
WorkerIngressMasterIpsecNat	IPsec NAT-T 数据包	**udp**	**4500**
WorkerIngressIpsecEsp	IPsec ESP 数据包	**50**	**All**
WorkerIngressMasterIpsecEsp	IPsec ESP 数据包	**50**	**All**
WorkerIngressInternalUDP	内部集群通信	**udp**	**9000 - 9999**
WorkerIngressMasterInternalUDP	内部集群通信	**udp**	**9000 - 9999**
WorkerIngressIngressServicesUDP	Kubernetes 入口服务	**udp**	**30000 - 32767**
WorkerIngressMasterIngressServicesUDP	Kubernetes 入口服务	**udp**	**30000 - 32767**

角色和实例配置集

您必须在 AWS 中为机器授予权限。提供的 CloudFormation 模板为以下 **AWS::IAM::Role** 对象授予机器 **Allow** 权限，并为每一组角色提供一个 **AWS::IAM::InstanceProfile**。如果不使用模板，您可以为机器授予以下宽泛权限或单独权限。

角色	影响	操作	资源
Master	**Allow**	**ec2:***	*
	Allow	**elasticloadbalancing:***	*
	Allow	**iam:PassRole**	*
	Allow	**s3:GetObject**	*
Worker	**Allow**	**ec2:Describe***	*
bootstrap	**Allow**	**ec2:Describe***	*
	Allow	**ec2:AttachVolume**	*
	Allow	**ec2:DetachVolume**	*

2.11.4.3. 证书签名请求管理

在使用您置备的基础架构时，集群只能有限地访问自动机器管理，因此您必须提供一种在安装后批准集群证书签名请求 (CSR) 的机制。**kube-controller-manager** 只能批准 kubelet 客户端 CSR。 **machine-approver** 无法保证使用 kubelet 凭证请求的提供证书的有效性，因为它不能确认是正确的机器发出了该请求。您必须决定并实施一种方法，以验证 kubelet 提供证书请求的有效性并进行批准。

2.11.4.4. 所需的 AWS 权限

注意

您的 IAM 用户必须具有 region **us-east-1** s 中的 permissions **tag:GetResources** 才能删除基本集群资源。作为 AWS API 要求的一部分，OpenShift Container Platform 安装程序在此区域中执行各种操作。

将 **AdministratorAccess** 策略附加到您在 Amazon Web Services (AWS) 中创建的 IAM 用户时，授予该用户所有需要的权限。要部署 OpenShift Container Platform 集群的所有组件，IAM 用户需要以下权限：

例 2.31. 安装所需的 EC2 权限

- **tag:TagResources**
- **tag:UntagResources**
- **ec2:AllocateAddress**
- **ec2:AssociateAddress**
- **ec2:AuthorizeSecurityGroupEgress**
- **ec2:AuthorizeSecurityGroupIngress**

- **ec2:CopyImage**
- **ec2:CreateNetworkInterface**
- **ec2:AttachNetworkInterface**
- **ec2:CreateSecurityGroup**
- **ec2:CreateTags**
- **ec2:CreateVolume**
- **ec2:DeleteSecurityGroup**
- **ec2:DeleteSnapshot**
- **ec2:DeleteTags**
- **ec2:DeregisterImage**
- **ec2:DescribeAccountAttributes**
- **ec2:DescribeAddresses**
- **ec2:DescribeAvailabilityZones**
- **ec2:DescribeDhcpOptions**
- **ec2:DescribeImages**
- **ec2:DescribeInstanceAttribute**
- **ec2:DescribeInstanceCreditSpecifications**
- **ec2:DescribeInstances**
- **ec2:DescribeInternetGateways**
- **ec2:DescribeKeyPairs**
- **ec2:DescribeNatGateways**
- **ec2:DescribeNetworkAcls**
- **ec2:DescribeNetworkInterfaces**
- **ec2:DescribePrefixLists**
- **ec2:DescribeRegions**
- **ec2:DescribeRouteTables**
- **ec2:DescribeSecurityGroups**
- **ec2:DescribeSubnets**
- **ec2:DescribeTags**

- **ec2:DescribeVolumes**
- **ec2:DescribeVpcAttribute**
- **ec2:DescribeVpcClassicLink**
- **ec2:DescribeVpcClassicLinkDnsSupport**
- **ec2:DescribeVpcEndpoints**
- **ec2:DescribeVpcs**
- **ec2:GetEbsDefaultKmsKeyId**
- **ec2:ModifyInstanceAttribute**
- **ec2:ModifyNetworkInterfaceAttribute**
- **ec2:ReleaseAddress**
- **ec2:RevokeSecurityGroupEgress**
- **ec2:RevokeSecurityGroupIngress**
- **ec2:RunInstances**
- **ec2:TerminateInstances**

例 2.32. 安装过程中创建网络资源所需的权限

- **ec2:AssociateDhcpOptions**
- **ec2:AssociateRouteTable**
- **ec2:AttachInternetGateway**
- **ec2:CreateDhcpOptions**
- **ec2:CreateInternetGateway**
- **ec2:CreateNatGateway**
- **ec2:CreateRoute**
- **ec2:CreateRouteTable**
- **ec2:CreateSubnet**
- **ec2:CreateVpc**
- **ec2:CreateVpcEndpoint**
- **ec2:ModifySubnetAttribute**
- **ec2:ModifyVpcAttribute**

注意

如果您使用现有的 VPC，您的帐户不需要这些权限来创建网络资源。

例 2.33. 安装所需的 Elastic Load Balancing 权限(ELB)

- **elasticloadbalancing:AddTags**
- **elasticloadbalancing:ApplySecurityGroupsToLoadBalancer**
- **elasticloadbalancing:AttachLoadBalancerToSubnets**
- **elasticloadbalancing:ConfigureHealthCheck**
- **elasticloadbalancing:CreateLoadBalancer**
- **elasticloadbalancing:CreateLoadBalancerListeners**
- **elasticloadbalancing:DeleteLoadBalancer**
- **elasticloadbalancing:DeregisterInstancesFromLoadBalancer**
- **elasticloadbalancing:DescribeInstanceHealth**
- **elasticloadbalancing:DescribeLoadBalancerAttributes**
- **elasticloadbalancing:DescribeLoadBalancers**
- **elasticloadbalancing:DescribeTags**
- **elasticloadbalancing:ModifyLoadBalancerAttributes**
- **elasticloadbalancing:RegisterInstancesWithLoadBalancer**
- **elasticloadbalancing:SetLoadBalancerPoliciesOfListener**

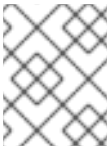
例 2.34. 安装所需的 Elastic Load Balancing 权限(ELBv2)

- **elasticloadbalancing:AddTags**
- **elasticloadbalancing:CreateListener**
- **elasticloadbalancing:CreateLoadBalancer**
- **elasticloadbalancing:CreateTargetGroup**
- **elasticloadbalancing:DeleteLoadBalancer**
- **elasticloadbalancing:DeregisterTargets**
- **elasticloadbalancing:DescribeListeners**
- **elasticloadbalancing:DescribeLoadBalancerAttributes**
- **elasticloadbalancing:DescribeLoadBalancers**

- **elasticloadbalancing:DescribeTargetGroupAttributes**
- **elasticloadbalancing:DescribeTargetHealth**
- **elasticloadbalancing:ModifyLoadBalancerAttributes**
- **elasticloadbalancing:ModifyTargetGroup**
- **elasticloadbalancing:ModifyTargetGroupAttributes**
- **elasticloadbalancing:RegisterTargets**

例 2.35. 安装所需的 IAM 权限

- **iam:AddRoleToInstanceProfile**
- **iam:CreateInstanceProfile**
- **iam:CreateRole**
- **iam:DeleteInstanceProfile**
- **iam:DeleteRole**
- **iam:DeleteRolePolicy**
- **iam:GetInstanceProfile**
- **iam:GetRole**
- **iam:GetRolePolicy**
- **iam:GetUser**
- **iam:ListInstanceProfilesForRole**
- **iam:ListRoles**
- **iam:ListUsers**
- **iam:PassRole**
- **iam:PutRolePolicy**
- **iam:RemoveRoleFromInstanceProfile**
- **iam:SimulatePrincipalPolicy**
- **iam:TagRole**



注意

如果您还没有在 AWS 帐户中创建弹性负载均衡器（ELB），IAM 用户还需要 **iam:CreateServiceLinkedRole** 权限。

例 2.36. 安装所需的 Route 53 权限

- **route53:ChangeResourceRecordSets**
- **route53:ChangeTagsForResource**
- **route53:CreateHostedZone**
- **route53:DeleteHostedZone**
- **route53:GetChange**
- **route53:GetHostedZone**
- **route53:ListHostedZones**
- **route53:ListHostedZonesByName**
- **route53:ListResourceRecordSets**
- **route53:ListTagsForResource**
- **route53:UpdateHostedZoneComment**

例 2.37. 安装所需的 S3 权限

- **s3:CreateBucket**
- **s3:DeleteBucket**
- **s3:GetAccelerateConfiguration**
- **s3:GetBucketAcl**
- **s3:GetBucketCors**
- **s3:GetBucketLocation**
- **s3:GetBucketLogging**
- **s3:GetBucketObjectLockConfiguration**
- **s3:GetBucketReplication**
- **s3:GetBucketRequestPayment**
- **s3:GetBucketTagging**
- **s3:GetBucketVersioning**
- **s3:GetBucketWebsite**
- **s3:GetEncryptionConfiguration**
- **s3:GetLifecycleConfiguration**

- **s3:GetReplicationConfiguration**
- **s3:ListBucket**
- **s3:PutBucketAcl**
- **s3:PutBucketTagging**
- **s3:PutEncryptionConfiguration**

例 2.38. 集群 Operators 所需的 S3 权限

- **s3:DeleteObject**
- **s3:GetObject**
- **s3:GetObjectAcl**
- **s3:GetObjectTagging**
- **s3:GetObjectVersion**
- **s3:PutObject**
- **s3:PutObjectAcl**
- **s3:PutObjectTagging**

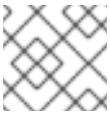
例 2.39. 删除基本集群资源所需的权限

- **autoscaling:DescribeAutoScalingGroups**
- **ec2:DeleteNetworkInterface**
- **ec2:DeleteVolume**
- **elasticloadbalancing:DeleteTargetGroup**
- **elasticloadbalancing:DescribeTargetGroups**
- **iam:DeleteAccessKey**
- **iam:DeleteUser**
- **iam:ListAttachedRolePolicies**
- **iam:ListInstanceProfiles**
- **iam:ListRolePolicies**
- **iam:ListUserPolicies**
- **s3:DeleteObject**
- **s3:ListBucketVersions**

- **tag:GetResources**

例 2.40. 删除网络资源所需的权限

- **ec2:DeleteDhcpOptions**
- **ec2:DeleteInternetGateway**
- **ec2:DeleteNatGateway**
- **ec2:DeleteRoute**
- **ec2:DeleteRouteTable**
- **ec2:DeleteSubnet**
- **ec2:DeleteVpc**
- **ec2:DeleteVpcEndpoints**
- **ec2:DetachInternetGateway**
- **ec2:DisassociateRouteTable**
- **ec2:ReplaceRouteTableAssociation**



注意

如果您使用现有的 VPC，您的帐户不需要这些权限来删除网络资源。

例 2.41. 创建清单所需的额外 IAM 和 S3 权限

- **iam:DeleteAccessKey**
- **iam:DeleteUser**
- **iam:DeleteUserPolicy**
- **iam:GetUserPolicy**
- **iam:ListAccessKeys**
- **iam:PutUserPolicy**
- **iam:TagUser**
- **iam:GetUserPolicy**
- **iam:ListAccessKeys**
- **s3:PutBucketPublicAccessBlock**
- **s3:GetBucketPublicAccessBlock**

- **s3:PutLifecycleConfiguration**
- **s3:HeadBucket**
- **s3:ListBucketMultipartUploads**
- **s3:AbortMultipartUpload**

注意

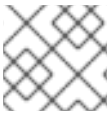
如果您要使用 mint 模式管理云供应商凭证，IAM 用户还需要 The **iam:CreateAccessKey and iam:CreateUser** 权限。

例 2.42. 安装时配额检查的可选权限

- **servicequotas:ListAWSDefaultServiceQuotas**

2.11.5. 生成 SSH 私钥并将其添加到代理中

如果要在集群上执行安装调试或灾难恢复，则必须为 **ssh-agent** 和安装程序提供 SSH 密钥。您可以使用此密钥访问公共集群中的 bootstrap 机器来排除安装问题。



注意

在生产环境中，您需要进行灾难恢复和调试。

您可以使用此密钥以 **core** 用户身份通过 SSH 连接到 master 节点。在部署集群时，此密钥会添加到 **core** 用户的 **~/.ssh/authorized_keys** 列表中。

注意

您必须使用一个本地密钥，而不要使用在特定平台上配置的密钥，如 AWS 密钥对。

流程

1. 如果还没有为计算机上免密码身份验证而配置的 SSH 密钥，请创建一个。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ ssh-keygen -t ed25519 -N '' \
  -f <path>/<file_name> 1
```

1 指定新 SSH 密钥的路径和文件名，如 **~/.ssh/id_rsa**。如果您已有密钥对，请确保您的公钥位于 **~/.ssh** 目录中。

运行此命令会在指定的位置生成不需要密码的 SSH 密钥。

注意

如果您计划在 **x86_64** 架构中安装使用 FIPS 验证的/Modules in Process 加密库的 OpenShift Container Platform 集群，不要创建使用 **ed25519** 算法的密钥。反之，创建一个使用 **rsa** 或 **ecdsa** 算法的密钥。

2. 作为后台任务启动 **ssh-agent** 进程：

```
$ eval "$(ssh-agent -s)"
```

输出示例

```
Agent pid 31874
```

注意

如果您的集群采用 FIPS 模式，则只使用 FIPS 兼容算法来生成 SSH 密钥。密钥必须是 RSA 或 ECDSA。

3. 将 SSH 私钥添加到 **ssh-agent**：

```
$ ssh-add <path>/<file_name> (1)
```

输出示例

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

(1) 指定 SSH 私钥的路径和文件名，如 **~/.ssh/id_rsa**

后续步骤

- 在安装 OpenShift Container Platform 时，为安装程序提供 SSH 公钥。如果在您置备的基础架构上安装集群，您必须将此密钥提供给集群的机器。

2.11.6. 创建用于 AWS 的安装文件

要使用用户置备的基础架构在 Amazon Web Services (AWS) 上安装 OpenShift Container Platform，您必须生成并修改安装程序部署集群所需的文件，以便集群只创建要使用的机器。您要生成并自定义 **install-config.yaml** 文件、Kubernetes 清单和 Ignition 配置文件。您也可以选择在安装准备阶段首先设置独立的 **var** 分区。

2.11.6.1. 可选：创建独立 /var 分区

建议安装程序将 OpenShift Container Platform 的磁盘分区保留给安装程序。然而，在有些情况下您可能需要在文件系统的一部分中创建独立分区。

OpenShift Container Platform 支持添加单个分区来将存储附加到 **/var** 分区或 **/var** 的子目录。例如：

- **/var/lib/containers**：保存镜像相关的内容，随着更多镜像和容器添加到系统中，它所占用的存储会增加。

- **/var/lib/etcd**：保存您可能希望保持独立的数据，比如 etcd 存储的性能优化。
- **/var**：保存您希望独立保留的数据，用于特定目的（如审计）。

单独存储 **/var** 目录的内容可方便地根据需要对区域扩展存储，并可以在以后重新安装 OpenShift Container Platform 时保持该数据地完整。使用这个方法，您不必再次拉取所有容器，在更新系统时也无法复制大量日志文件。

因为 **/var** 在进行一个全新的 Red Hat Enterprise Linux CoreOS（RHCOS）安装前必需存在，所以这个流程会在 OpenShift Container Platform 安装过程的 **openshift-install** 准备阶段插入的机器配置来设置独立的 **/var** 分区。

重要

如果按照以下步骤在此流程中创建独立 **/var** 分区，则不需要再次创建 Kubernetes 清单和 Ignition 配置文件，如本节所述。

流程

1. 创建存放 OpenShift Container Platform 安装文件的目录：

```
$ mkdir $HOME/clusterconfig
```

2. 运行 **openshift-install** 在 **manifest** 和 **openshift** 子目录中创建一组文件。在出现提示时回答系统问题：

```
$ openshift-install create manifests --dir $HOME/clusterconfig
```

输出示例

```
? SSH Public Key ...
INFO Credentials loaded from the "myprofile" profile in file "/home/myuser/.aws/credentials"
INFO Consuming Install Config from target directory
INFO Manifests created in: $HOME/clusterconfig/manifests and
$HOME/clusterconfig/openshift
```

3. 可选：确认安装程序在 **clusterconfig/openshift** 目录中创建了清单：

```
$ ls $HOME/clusterconfig/openshift/
```

输出示例

```
99_kubeadmin-password-secret.yaml
99_openshift-cluster-api_master-machines-0.yaml
99_openshift-cluster-api_master-machines-1.yaml
99_openshift-cluster-api_master-machines-2.yaml
...
```

4. 创建 **MachineConfig** 对象并将其添加到 **openshift** 目录中的一个文件中。例如，把文件命名为 **98-var-partition.yaml**，将磁盘设备名称改为 **worker** 系统中存储设备的名称，并根据情况设置存储大小。这个示例将 **/var** 目录放在一个单独的分区中：

```
apiVersion: machineconfiguration.openshift.io/v1
```

```
kind: MachineConfig
metadata:
  labels:
    machineconfiguration.openshift.io/role: worker
  name: 98-var-partition
spec:
  config:
    ignition:
      version: 3.1.0
    storage:
      disks:
      - device: /dev/<device_name> 1
        partitions:
        - label: var
          startMiB: <partition_start_offset> 2
          sizeMiB: <partition_size> 3
      filesystems:
        - device: /dev/disk/by-partlabel/var
          path: /var
          format: xfs
    systemd:
      units:
        - name: var.mount 4
          enabled: true
          contents: |
            [Unit]
            Before=local-fs.target
            [Mount]
            What=/dev/disk/by-partlabel/var
            Where=/var
            Options=defaults,prjquota 5
            [Install]
            WantedBy=local-fs.target
```

1. 要分区的磁盘的存储设备名称。
2. 当在引导磁盘中添加数据分区时，推荐最少使用 25000 MiB（Mebibytes）。root 文件系统会自动重新定义大小使其占据所有可用空间（最多到指定的偏移值）。如果没有指定值，或者指定的值小于推荐的最小值，则生成的 root 文件系统会太小，而在以后进行的 RHCOS 重新安装可能会覆盖数据分区的开始部分。
3. 数据分区的大小（以兆字节为单位）。
4. 挂载单元的名称必须与 **Where=** 指令中指定的目录匹配。例如，对于挂载在 **/var/lib/containers** 上的文件系统，该单元必须命名为 **var-lib-containers.mount**。
5. 对于用于容器存储的文件系统，必须启用 **prjquota** 挂载选项。

注意

在创建独立 **/var** 分区时，如果不同的实例类型没有相同的设备名称，则无法将不同的实例类型用于 worker 节点。

5. 再次运行 **openshift-install**，从 **manifest** 和 **openshift** 子目录中的一组文件创建 Ignition 配置：

```
$ openshift-install create ignition-configs --dir $HOME/clusterconfig
$ ls $HOME/clusterconfig/
auth  bootstrap.ign  master.ign  metadata.json  worker.ign
```

现在，您可以使用 Ignition 配置文件作为安装程序的输入来安装 Red Hat Enterprise Linux CoreOS（RHCOS）系统。

2.11.6.2. 创建安装配置文件

生成并自定义安装程序部署集群所需的安装配置文件。

先决条件

- 已获取 OpenShift Container Platform 安装程序用于用户置备的基础架构和集群的 pull secret。对于受限网络安装，这些文件位于您的堡垒主机上。
- 使用红帽发布的附带 Red Hat Enterprise Linux CoreOS（RHCOS）AMI 检查您是否将集群部署到一个区域。如果您要部署到需要自定义 AMI 的区域，如 AWS GovCloud 区域，您必须手动创建 **install-config.yaml** 文件。

流程

1. 创建 **install-config.yaml** 文件。
   a. 更改到包含安装程序的目录，再运行以下命令：

```
$ ./openshift-install create install-config --dir <installation_directory> 1
```

   1 对于 **<installation_directory>**，请指定用于保存安装程序所创建的文件的目录名称。

重要

指定一个空目录。一些安装信息，如 bootstrap X.509 证书，有较短的过期间隔，因此不要重复使用安装目录。如果要重复使用另一个集群安装中的个别文件，可以将其复制到您的目录中。但是，一些安装数据的文件名可能会在发行版本之间有所改变。从 OpenShift Container Platform 老版本中复制安装文件时要格外小心。

   b. 在提示符处，提供您的云的配置详情：
      i. 可选：选择用来访问集群机器的 SSH 密钥。

注意

对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 **ssh-agent** 进程使用的 SSH 密钥。

      ii. 选择 **aws** 作为目标平台。
      iii. 如果计算机上没有保存 AWS 配置集，请为您配置用于运行安装程序的用户输入 AWS 访问密钥 ID 和 secret 访问密钥。

注意

AWS 访问密钥 ID 和 secret 访问密钥存储在安装主机上当前用户主目录中的 **~/.aws/credentials** 中。如果文件中不存在导出的配置集凭证，安装程序会提示您输入凭证。您向安装程序提供的所有凭证都存储在文件中。

iv. 选择要将集群部署到的 AWS 区域。

v. 选择您为集群配置的 Route 53 服务的基域。

vi. 为集群输入一个描述性名称。

vii. 粘贴 Red Hat OpenShift Cluster Manager 中的 pull secret 。

2. 编辑 **install-config.yaml** 文件，以提供在受限网络中安装所需的其他信息。

a. 更新 **pullSecret** 值，使其包含 registry 的身份验证信息：

```
pullSecret: '{"auths":{"<local_registry>": {"auth": "<credentials>","email":
"you@example.com"}}}'
```

对于 **<local_registry>**，请指定 registry 域名，以及您的镜像 registry 用来提供内容的可选端口。例如： **registry.example.com** 或者 **registry.example.com:5000**。使用 **<credentials>** 为您生成的镜像 registry 指定 base64 编码的用户名和密码。

b. 添加 **additionalTrustBundle** 参数和值。该值必须是您用于镜像 registry 的证书文件内容，可以是现有的可信证书颁发机构或您为镜像 registry 生成的自签名证书。

```
additionalTrustBundle: |
  -----BEGIN CERTIFICATE-----

ZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZ
  -----END CERTIFICATE-----
```

c. 添加镜像内容资源：

```
imageContentSources:
- mirrors:
  - <local_registry>/<local_repository_name>/release
  source: quay.io/openshift-release-dev/ocp-release
- mirrors:
  - <local_registry>/<local_repository_name>/release
  source: quay.io/openshift-release-dev/ocp-v4.0-art-dev
```

使用命令输出中的 **imageContentSources** 部分来镜像（mirror）仓库，或您从您进入受限网络的介质中的内容时使用的值。

d. 可选：将发布策略设置为 **Internal**：

```
publish: Internal
```

通过设置这个选项，您可以创建一个内部 Ingress Controller 和一个私有负载均衡器。

3. 可选：备份 **install-config.yaml** 文件。

重要

install-config.yaml 文件会在安装过程中消耗掉。如果要重复使用此文件，必须现在备份。

其他资源

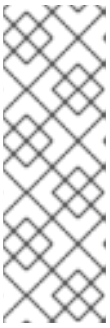
- 如需有关 AWS 配置集和凭证配置的更多信息，请参阅 AWS 文档中的配置和凭证文件设置。

2.11.6.3. 在安装过程中配置集群范围代理

生产环境可能会拒绝直接访问互联网，而是提供 HTTP 或 HTTPS 代理。您可以通过在 **install-config.yaml** 文件中配置代理设置，将新的 OpenShift Container Platform 集群配置为使用代理。

先决条件

- 您有一个现有的 **install-config.yaml** 文件。
- 您检查了集群需要访问的站点，并决定是否需要绕过代理。默认情况下代理所有集群出口流量，包括对托管云供应商 API 的调用。您需要将站点添加到 **Proxy** 对象的 **spec.noProxy** 字段来绕过代理。



注意

Proxy 对象 **status.noProxy** 字段使用安装配置中的 **networking.machineNetwork[].cidr**、**networking.clusterNetwork[].cidr** 和 **networking.serviceNetwork[]** 字段的值填充。

对于在 Amazon Web Services(AWS)、Google Cloud Platform(GCP)、Microsoft Azure 和 Red Hat OpenStack Platform(RHOSP)上安装, **Proxy** 对象 **status.noProxy** 字段也会使用实例元数据端点填充(**169.254.169.254**)。

- 如果您的集群位于 AWS 上，请将 **ec2.<region>.amazonaws.com**、**elasticloadbalancing.<region>.amazonaws.com** 和 **s3.<region>.amazonaws.com** 端点添加到 VPC 端点。需要这些端点才能完成节点到 AWS EC2 API 的请求。由于代理在容器级别而不是节点级别工作，因此您必须通过 AWS 专用网络将这些请求路由到 AWS EC2 API。在代理服务器中的允许列表中添加 EC2 API 的公共 IP 地址是不够的。

流程

1. 编辑 **install-config.yaml** 文件并添加代理设置。例如：

```
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: example.com 3
additionalTrustBundle: | 4
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
...
```

1. 用于创建集群外 HTTP 连接的代理 URL。URL 必须是 **http**。
2. 用于创建集群外 HTTPS 连接的代理 URL。
3. 要排除在代理中的目标域名、IP 地址或其他网络 CIDR 的逗号分隔列表。在域前面加 **.** 来仅匹配子域。例如： **.y.com** 匹配 **x.y.com**，但不匹配 **y.com**。使用 ***** 绕过所有目的地的代理。
4. 如果提供，安装程序会在 **openshift-config 命名空间中生成名为 user-ca- bundle** 的配置映射来保存额外的 CA 证书。如果您提供 **additionalTrustBundle** 和至少一个代理设置，则 **Proxy** 对象会被配置为引用 **trustedCA** 字段中的 **user-ca-bundle** 配置映射。然后，Cluster Network Operator 会创建一个 **trusted-ca-bundle** 配置映射，该配置映射将为 **trustedCA** 参数指定的内容与 RHCOS 信任捆绑包合并。**additionalTrustBundle** 字段是必需的，除非代理的身份证书由来自 RHCOS 信任捆绑包的颁发机构签名。

注意

安装程序不支持代理的 **readinessEndpoints** 字段。

2. 保存该文件，并在安装 OpenShift Container Platform 时引用。

安装程序会创建一个名为 **cluster** 的集群范围代理，该代理使用提供的 **install-config.yaml** 文件中的代理设置。如果没有提供代理设置，仍然会创建一个 **cluster Proxy** 对象，但它会有一个空 **spec**。

注意

只支持名为 **cluster** 的 **Proxy** 对象，且无法创建额外的代理。

2.11.6.4. 创建 Kubernetes 清单和 Ignition 配置文件

由于您必须修改一些集群定义文件并要手动启动集群机器，因此您必须生成 Kubernetes 清单和 Ignition 配置文件，集群需要这两项来创建其机器。

安装配置文件转换为 Kubernetes 清单。清单嵌套到 Ignition 配置文件中，稍后用于创建集群。

重要

- 安装程序生成的 Ignition 配置文件包含在 24 小时后过期的证书，然后在过期时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外情况是，您需要手动批准待处理的 **node-bootstrapper** 证书签名请求（CSR）来恢复 kubelet 证书。如需更多信息，请参阅 *从过期的 control plane 证书中恢复* 的文档。
- 建议您在生成 12 小时后使用 Ignition 配置文件，因为集群安装后 24 小时证书从 16 小时轮转至 22 小时。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中运行证书更新时避免安装失败。

先决条件

- 已获得 OpenShift Container Platform 安装程序。对于受限网络安装，这些文件位于您的堡垒主机上。
- 已创建 **install-config.yaml** 安装配置文件。

流程

1. 切换到包含安装程序的目录，并为集群生成 Kubernetes 清单：

```
$ ./openshift-install create manifests --dir <installation_directory> 1
```

1 对于 **<installation_directory>**，请指定含有您创建的 **install-config.yaml** 文件的安装目录。

2. 删除定义 control plane 机器的 Kubernetes 清单文件：

```
$ rm -f <installation_directory>/openshift/99_openshift-cluster-api_master-machines-*.yaml
```

通过删除这些文件，您可以防止集群自动生成 control plane 机器。

3. 删除定义 worker 机器的 Kubernetes 清单文件：

```
$ rm -f <installation_directory>/openshift/99_openshift-cluster-api_worker-machineset-*.yaml
```

由于您要自行创建并管理 worker 机器，因此不需要初始化这些机器。

4. 检查 **<installation_directory>/manifests/cluster-scheduler-02-config.yml** Kubernetes 清单文件中的 **mastersSchedulable** 参数是否已设置为 **false**。此设置可防止在 control plane 机器上调度 pod:

   a. 打开 **<installation_directory>/manifests/cluster-scheduler-02-config.yml** 文件。

   b. 找到 **mastersSchedulable** 参数并确保它被设置为 **false**。

   c. 保存并退出文件。

5. 可选：如果您不希望 Ingress Operator 代表您创建 DNS 记录，请删除 **<installation_directory>/manifests/cluster-dns-02-config.yml** DNS 配置文件中的 **privateZone** 和 **publicZone** 部分：

```
apiVersion: config.openshift.io/v1
kind: DNS
metadata:
  creationTimestamp: null
  name: cluster
spec:
  baseDomain: example.openshift.com
  privateZone: 1
    id: mycluster-100419-private-zone
  publicZone: 2
    id: example.openshift.com
status: {}
```

1 2 完全删除此部分。

如果您这样做，后续步骤中必须手动添加入口 DNS 记录。

6. 要创建 Ignition 配置文件，从包含安装程序的目录运行以下命令：

```
$ ./openshift-install create ignition-configs --dir <installation_directory> 1
```

1 对于 **<installation_directory>**，请指定相同的安装目录。

该目录中将生成以下文件：

```
.
├── auth
│   ├── kubeadmin-password
│   └── kubeconfig
├── bootstrap.ign
├── master.ign
├── metadata.json
└── worker.ign
```

2.11.7. 提取基础架构名称

Ignition 配置文件包含一个唯一集群标识符，您可以使用它在 Amazon Web Services (AWS) 中唯一地标识您的集群。基础架构名称还用于在 OpenShift Container Platform 安装过程中定位适当的 AWS 资源。提供的 CloudFormation 模板包含对此基础架构名称的引用，因此您必须提取它。

先决条件

- 获取 OpenShift Container Platform 安装程序以及集群的 pull secret。
- 已为集群生成 Ignition 配置文件。
- 安装了 **jq** 软件包。

流程

- 要从 Ignition 配置文件元数据中提取和查看基础架构名称，请运行以下命令：

```
$ jq -r .infraID <installation_directory>/metadata.json 1
```

1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

输出示例

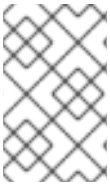
```
openshift-vw9j6 1
```

1 此命令的输出是您的集群名称和随机字符串。

2.11.8. 在 AWS 中创建 VPC

您必须在 Amazon Web Services（AWS）中创建 Virtual Private Cloud（VPC），供您的 OpenShift Container Platform 集群使用。您可以自定义 VPC 来满足您的要求，包括 VPN 和路由表。

您可以使用提供的 CloudFormation 模板和自定义参数文件创建代表 VPC 的 AWS 资源堆栈。



注意

如果不使用提供的 CloudFormation 模板来创建 AWS 基础架构，您必须检查提供的信息并手动创建基础架构。如果集群没有正确初始化，您可能需要联系红帽支持并提供您的安装日志。

先决条件

- 已配置了一个 AWS 帐户。
- 您可以通过运行 **aws configure**，将 AWS 密钥和区域添加到本地 AWS 配置集中。
- 已为集群生成 Ignition 配置文件。

流程

1. 创建一个 JSON 文件，其包含模板所需的参数值：

```
[
  {
    "ParameterKey": "VpcCidr", 1
    "ParameterValue": "10.0.0.0/16" 2
  },
  {
    "ParameterKey": "AvailabilityZoneCount", 3
    "ParameterValue": "1" 4
  },
  {
    "ParameterKey": "SubnetBits", 5
    "ParameterValue": "12" 6
  }
]
```

1. VPC 的 CIDR 块。
2. 以 **x.x.x.x/16-24** 格式指定 CIDR 块。
3. 在其中部署 VPC 的可用区的数量。
4. 指定一个 **1** 到 **3** 之间的整数。
5. 各个可用区中每个子网的大小。
6. 指定 **5** 到 **13** 之间的整数，其中 **5** 为 **/27**，**13** 为 **/19**。

2. 复制本主题的 **VPC 的 CloudFormation 模板**部分中的模板，并将它以 YAML 文件形式保存到计算机上。此模板描述了集群所需的 VPC。
3. 启动 CloudFormation 模板，以创建代表 VPC 的 AWS 资源堆栈：

重要

您必须在一行内输入命令。

-

```
$ aws cloudformation create-stack --stack-name <name> 1
    --template-body file://<template>.yaml 2
    --parameters file://<parameters>.json 3
```

1 **<name>** 是 CloudFormation 堆栈的名称，如 **cluster-VPC**。如果您删除集群，则需要此堆栈的名称。

2 **<template>** 是您保存的 CloudFormation 模板 YAML 文件的相对路径和名称。

3 **<parameters>** 是 CloudFormation 参数 JSON 文件的相对路径和名称。

输出示例

```
arn:aws:cloudformation:us-east-1:269333783861:stack/cluster-vpc/dbedae40-2fd3-11eb-820e-12a48460849f
```

4. 确认模板组件已存在：

```
$ aws cloudformation describe-stacks --stack-name <name>
```

在 **StackStatus** 显示 **CREATE_COMPLETE** 后，输出会显示以下参数的值。您必须将这些参数值提供给您在创建集群时要运行的其他 CloudFormation 模板：

VpcId	您的 VPC ID。
PublicSubnetIds	新公共子网的 ID。
PrivateSubnetIds	新专用子网的 ID。

2.11.8.1. VPC 的 CloudFormation 模板

您可以使用以下 CloudFormation 模板来部署 OpenShift Container Platform 集群所需的 VPC。

例 2.43. VPC 的 CloudFormation 模板

```
AWSTemplateFormatVersion: 2010-09-09
Description: Template for Best Practice VPC with 1-3 AZs

Parameters:
  VpcCidr:
    AllowedPattern: ^(([0-9]|[1-9][0-9]|1[0-9]{2}|2[0-4][0-9]|25[0-5])\.){3}([0-9]|[1-9][0-9]|1[0-9]{2}|2[0-4][0-9]|25[0-5])(\/(1[6-9]|2[0-4]))$
    ConstraintDescription: CIDR block parameter must be in the form x.x.x.x/16-24.
    Default: 10.0.0.0/16
    Description: CIDR block for VPC.
    Type: String
  AvailabilityZoneCount:
    ConstraintDescription: "The number of availability zones. (Min: 1, Max: 3)"
    MinValue: 1
```

```
    MaxValue: 3
    Default: 1
    Description: "How many AZs to create VPC subnets for. (Min: 1, Max: 3)"
    Type: Number
  SubnetBits:
    ConstraintDescription: CIDR block parameter must be in the form x.x.x.x/19-27.
    MinValue: 5
    MaxValue: 13
    Default: 12
    Description: "Size of each subnet to create within the availability zones. (Min: 5 = /27, Max: 13 =
/19)"
    Type: Number

Metadata:
  AWS::CloudFormation::Interface:
    ParameterGroups:
    - Label:
        default: "Network Configuration"
      Parameters:
      - VpcCidr
      - SubnetBits
    - Label:
        default: "Availability Zones"
      Parameters:
      - AvailabilityZoneCount
    ParameterLabels:
      AvailabilityZoneCount:
        default: "Availability Zone Count"
      VpcCidr:
        default: "VPC CIDR"
      SubnetBits:
        default: "Bits Per Subnet"

Conditions:
  DoAz3: !Equals [3, !Ref AvailabilityZoneCount]
  DoAz2: !Or [!Equals [2, !Ref AvailabilityZoneCount], Condition: DoAz3]

Resources:
  VPC:
    Type: "AWS::EC2::VPC"
    Properties:
      EnableDnsSupport: "true"
      EnableDnsHostnames: "true"
      CidrBlock: !Ref VpcCidr
  PublicSubnet:
    Type: "AWS::EC2::Subnet"
    Properties:
      VpcId: !Ref VPC
      CidrBlock: !Select [0, !Cidr [!Ref VpcCidr, 6, !Ref SubnetBits]]
      AvailabilityZone: !Select
      - 0
      - Fn::GetAZs: !Ref "AWS::Region"
  PublicSubnet2:
    Type: "AWS::EC2::Subnet"
    Condition: DoAz2
    Properties:
```

```
      VpcId: !Ref VPC
      CidrBlock: !Select [1, !Cidr [!Ref VpcCidr, 6, !Ref SubnetBits]]
      AvailabilityZone: !Select
      - 1
      - Fn::GetAZs: !Ref "AWS::Region"
  PublicSubnet3:
    Type: "AWS::EC2::Subnet"
    Condition: DoAz3
    Properties:
      VpcId: !Ref VPC
      CidrBlock: !Select [2, !Cidr [!Ref VpcCidr, 6, !Ref SubnetBits]]
      AvailabilityZone: !Select
      - 2
      - Fn::GetAZs: !Ref "AWS::Region"
  InternetGateway:
    Type: "AWS::EC2::InternetGateway"
  GatewayToInternet:
    Type: "AWS::EC2::VPCGatewayAttachment"
    Properties:
      VpcId: !Ref VPC
      InternetGatewayId: !Ref InternetGateway
  PublicRouteTable:
    Type: "AWS::EC2::RouteTable"
    Properties:
      VpcId: !Ref VPC
  PublicRoute:
    Type: "AWS::EC2::Route"
    DependsOn: GatewayToInternet
    Properties:
      RouteTableId: !Ref PublicRouteTable
      DestinationCidrBlock: 0.0.0.0/0
      GatewayId: !Ref InternetGateway
  PublicSubnetRouteTableAssociation:
    Type: "AWS::EC2::SubnetRouteTableAssociation"
    Properties:
      SubnetId: !Ref PublicSubnet
      RouteTableId: !Ref PublicRouteTable
  PublicSubnetRouteTableAssociation2:
    Type: "AWS::EC2::SubnetRouteTableAssociation"
    Condition: DoAz2
    Properties:
      SubnetId: !Ref PublicSubnet2
      RouteTableId: !Ref PublicRouteTable
  PublicSubnetRouteTableAssociation3:
    Condition: DoAz3
    Type: "AWS::EC2::SubnetRouteTableAssociation"
    Properties:
      SubnetId: !Ref PublicSubnet3
      RouteTableId: !Ref PublicRouteTable
  PrivateSubnet:
    Type: "AWS::EC2::Subnet"
    Properties:
      VpcId: !Ref VPC
      CidrBlock: !Select [3, !Cidr [!Ref VpcCidr, 6, !Ref SubnetBits]]
      AvailabilityZone: !Select
      - 0
```

```
    - Fn::GetAZs: !Ref "AWS::Region"
PrivateRouteTable:
  Type: "AWS::EC2::RouteTable"
  Properties:
    VpcId: !Ref VPC
PrivateSubnetRouteTableAssociation:
  Type: "AWS::EC2::SubnetRouteTableAssociation"
  Properties:
    SubnetId: !Ref PrivateSubnet
    RouteTableId: !Ref PrivateRouteTable
NAT:
  DependsOn:
  - GatewayToInternet
  Type: "AWS::EC2::NatGateway"
  Properties:
    AllocationId:
      "Fn::GetAtt":
      - EIP
      - AllocationId
    SubnetId: !Ref PublicSubnet
EIP:
  Type: "AWS::EC2::EIP"
  Properties:
    Domain: vpc
Route:
  Type: "AWS::EC2::Route"
  Properties:
    RouteTableId:
      Ref: PrivateRouteTable
    DestinationCidrBlock: 0.0.0.0/0
    NatGatewayId:
      Ref: NAT
PrivateSubnet2:
  Type: "AWS::EC2::Subnet"
  Condition: DoAz2
  Properties:
    VpcId: !Ref VPC
    CidrBlock: !Select [4, !Cidr [!Ref VpcCidr, 6, !Ref SubnetBits]]
    AvailabilityZone: !Select
    - 1
    - Fn::GetAZs: !Ref "AWS::Region"
PrivateRouteTable2:
  Type: "AWS::EC2::RouteTable"
  Condition: DoAz2
  Properties:
    VpcId: !Ref VPC
PrivateSubnetRouteTableAssociation2:
  Type: "AWS::EC2::SubnetRouteTableAssociation"
  Condition: DoAz2
  Properties:
    SubnetId: !Ref PrivateSubnet2
    RouteTableId: !Ref PrivateRouteTable2
NAT2:
  DependsOn:
  - GatewayToInternet
  Type: "AWS::EC2::NatGateway"
```

```
  Condition: DoAz2
  Properties:
    AllocationId:
      "Fn::GetAtt":
      - EIP2
      - AllocationId
    SubnetId: !Ref PublicSubnet2
EIP2:
  Type: "AWS::EC2::EIP"
  Condition: DoAz2
  Properties:
    Domain: vpc
Route2:
  Type: "AWS::EC2::Route"
  Condition: DoAz2
  Properties:
    RouteTableId:
      Ref: PrivateRouteTable2
    DestinationCidrBlock: 0.0.0.0/0
    NatGatewayId:
      Ref: NAT2
PrivateSubnet3:
  Type: "AWS::EC2::Subnet"
  Condition: DoAz3
  Properties:
    VpcId: !Ref VPC
    CidrBlock: !Select [5, !Cidr [!Ref VpcCidr, 6, !Ref SubnetBits]]
    AvailabilityZone: !Select
    - 2
    - Fn::GetAZs: !Ref "AWS::Region"
PrivateRouteTable3:
  Type: "AWS::EC2::RouteTable"
  Condition: DoAz3
  Properties:
    VpcId: !Ref VPC
PrivateSubnetRouteTableAssociation3:
  Type: "AWS::EC2::SubnetRouteTableAssociation"
  Condition: DoAz3
  Properties:
    SubnetId: !Ref PrivateSubnet3
    RouteTableId: !Ref PrivateRouteTable3
NAT3:
  DependsOn:
  - GatewayToInternet
  Type: "AWS::EC2::NatGateway"
  Condition: DoAz3
  Properties:
    AllocationId:
      "Fn::GetAtt":
      - EIP3
      - AllocationId
    SubnetId: !Ref PublicSubnet3
EIP3:
  Type: "AWS::EC2::EIP"
  Condition: DoAz3
  Properties:
```

```
      Domain: vpc
  Route3:
    Type: "AWS::EC2::Route"
    Condition: DoAz3
    Properties:
      RouteTableId:
        Ref: PrivateRouteTable3
      DestinationCidrBlock: 0.0.0.0/0
      NatGatewayId:
        Ref: NAT3
  S3Endpoint:
    Type: AWS::EC2::VPCEndpoint
    Properties:
      PolicyDocument:
        Version: 2012-10-17
        Statement:
        - Effect: Allow
          Principal: '*'
          Action:
          - '*'
          Resource:
          - '*'
      RouteTableIds:
      - !Ref PublicRouteTable
      - !Ref PrivateRouteTable
      - !If [DoAz2, !Ref PrivateRouteTable2, !Ref "AWS::NoValue"]
      - !If [DoAz3, !Ref PrivateRouteTable3, !Ref "AWS::NoValue"]
      ServiceName: !Join
      - ''
      - - com.amazonaws.
        - !Ref 'AWS::Region'
        - .s3
      VpcId: !Ref VPC

Outputs:
  VpcId:
    Description: ID of the new VPC.
    Value: !Ref VPC
  PublicSubnetIds:
    Description: Subnet IDs of the public subnets.
    Value:
      !Join [
        ",",
        [!Ref PublicSubnet, !If [DoAz2, !Ref PublicSubnet2, !Ref "AWS::NoValue"], !If [DoAz3, !Ref
PublicSubnet3, !Ref "AWS::NoValue"]]
      ]
  PrivateSubnetIds:
    Description: Subnet IDs of the private subnets.
    Value:
      !Join [
        ",",
        [!Ref PrivateSubnet, !If [DoAz2, !Ref PrivateSubnet2, !Ref "AWS::NoValue"], !If [DoAz3, !Ref
PrivateSubnet3, !Ref "AWS::NoValue"]]
      ]
```

2.11.9. 在 AWS 中创建网络和负载均衡组件

您必须在 OpenShift Container Platform 集群可以使用的 Amazon Web Services（AWS）中配置网络、经典或网络负载均衡。

您可以使用提供的 CloudFormation 模板和自定义参数文件来创建 AWS 资源堆栈。堆栈代表 OpenShift Container Platform 集群所需的网络和负载均衡组件。该模板还创建一个托管区和子网标签。

您可以在单一虚拟私有云(VPC)内多次运行该模板。

注意

如果不使用提供的 CloudFormation 模板来创建 AWS 基础架构，您必须检查提供的信息并手动创建基础架构。如果集群没有正确初始化，您可能需要联系红帽支持并提供您的安装日志。

先决条件

- 已配置了一个 AWS 帐户。
- 您可以通过运行 **aws configure**，将 AWS 密钥和区域添加到本地 AWS 配置集中。
- 已为集群生成 Ignition 配置文件。
- 您在 AWS 中创建并配置了 VPC 及相关子网。

流程

1. 获取您在 **install-config.yaml** 文件中为集群指定的 Route 53 基域的托管区 ID。您可以运行以下命令来获取托管区的详细信息：

```
$ aws route53 list-hosted-zones-by-name --dns-name <route53_domain> 1
```

1 对于 **<route53_domain>**，请指定您为集群生成 **install-config.yaml** 文件时所用的 Route53 基域。

输出示例

```
mycluster.example.com. False 100
HOSTEDZONES 65F8F38E-2268-B835-E15C-AB55336FCBFA
/hostedzone/Z21IXYZABCZ2A4 mycluster.example.com. 10
```

在示例输出中，托管区 ID 为 **Z21IXYZABCZ2A4**。

2. 创建一个 JSON 文件，其包含模板所需的参数值：

```
[
  {
    "ParameterKey": "ClusterName", 1
    "ParameterValue": "mycluster" 2
  },
  {
    "ParameterKey": "InfrastructureName", 3
```

```
    "ParameterValue": "mycluster-<random_string>" 4
  },
  {
    "ParameterKey": "HostedZoneId", 5
    "ParameterValue": "<random_string>" 6
  },
  {
    "ParameterKey": "HostedZoneName", 7
    "ParameterValue": "example.com" 8
  },
  {
    "ParameterKey": "PublicSubnets", 9
    "ParameterValue": "subnet-<random_string>" 10
  },
  {
    "ParameterKey": "PrivateSubnets", 11
    "ParameterValue": "subnet-<random_string>" 12
  },
  {
    "ParameterKey": "VpcId", 13
    "ParameterValue": "vpc-<random_string>" 14
  }
]
```

1. 一个简短的、代表集群的名称用于主机名等。
2. 指定您为集群生成 **install-config.yaml** 文件时所用的集群名称。
3. 您的 Ignition 配置文件中为集群编码的集群基础架构名称。
4. 指定从 Ignition 配置文件元数据中提取的基础架构名称，其格式为 **<cluster-name>-<random-string>**。
5. 用来注册目标的 Route 53 公共区 ID。
6. 指定 Route 53 公共区 ID，其格式与 **Z21IXYZABCZ2A4** 类似。您可以从 AWS 控制台获取这个值。
7. 用来注册目标的 Route 53 区。
8. 指定您为集群生成 **install-config.yaml** 文件时所用的 Route 53 基域。请勿包含 AWS 控制台中显示的结尾句点 (.)。
9. 为 VPC 创建的公共子网。
10. 指定 VPC 的 CloudFormation 模板输出的 **PublicSubnetIds** 值。
11. 为 VPC 创建的专用子网。
12. 指定 VPC 的 CloudFormation 模板输出的 **PrivateSubnetIds** 值。
13. 为集群创建的 VPC。
14. 指定 VPC 的 CloudFormation 模板输出的 **VpcId** 值。

3. 复制本主题的**网络和负载均衡器的 CloudFormation 模板**部分中的模板，并将它以 YAML 文件形式保存到计算机上。此模板描述了集群所需的网络和负载均衡对象。

重要

如果要将集群部署到 AWS 政府区域，您必须更新 CloudFormation 模板中的 **InternalApiServerRecord**，以使用 **CNAME** 记录。AWS 政府区不支持 **ALIAS** 类型的记录。

4. 启动 CloudFormation 模板，以创建 AWS 资源堆栈，该堆栈提供网络和负载均衡组件：

重要

您必须在一行内输入命令。

```
$ aws cloudformation create-stack --stack-name <name> 1
    --template-body file://<template>.yaml 2
    --parameters file://<parameters>.json 3
    --capabilities CAPABILITY_NAMED_IAM 4
```

1. **<name>** 是 CloudFormation 堆栈的名称，如 **cluster-dns**。如果您删除集群，则需要此堆栈的名称。
2. **<template>** 是您保存的 CloudFormation 模板 YAML 文件的相对路径和名称。
3. **<parameters>** 是 CloudFormation 参数 JSON 文件的相对路径和名称。
4. 您必须明确声明 **CAPABILITY_NAMED_IAM** 功能，因为提供的模板会创建一些 **AWS::IAM::Role** 资源。

输出示例

```
arn:aws:cloudformation:us-east-1:269333783861:stack/cluster-dns/cd3e5de0-2fd4-11eb-5cf0-12be5c33a183
```

5. 确认模板组件已存在：

```
$ aws cloudformation describe-stacks --stack-name <name>
```

在 **StackStatus** 显示 **CREATE_COMPLETE** 后，输出会显示以下参数的值。您必须将这些参数值提供给您在创建集群时要运行的其他 CloudFormation 模板：

PrivateHostedZoneId	专用 DNS 的托管区 ID。
ExternalApiLoadBalancerName	外部 API 负载均衡器的完整名称。

InternalApiLoadBalancerName	内部 API 负载均衡器的完整名称。
ApiServerDnsName	API 服务器的完整主机名。
RegisterNlbIpTargetsLambda	有助于为这些负载均衡器注册/撤销注册 IP 目标的 Lambda ARN。
ExternalApiTargetGroupArn	外部 API 目标组的 ARN。
InternalApiTargetGroupArn	内部 API 目标组的 ARN。
InternalServiceTargetGroupArn	内部服务目标组群的 ARN。

2.11.9.1. 网络和负载均衡器的 CloudFormation 模板

您可以使用以下 CloudFormation 模板来部署 OpenShift Container Platform 集群所需的网络对象和负载均衡器。

例 2.44. 网络和负载均衡器的 CloudFormation 模板

```
AWSTemplateFormatVersion: 2010-09-09
Description: Template for OpenShift Cluster Network Elements (Route53 & LBs)

Parameters:
  ClusterName:
    AllowedPattern: ^([a-zA-Z][a-zA-Z0-9\-]{0,26})$
    MaxLength: 27
    MinLength: 1
    ConstraintDescription: Cluster name must be alphanumeric, start with a letter, and have a
maximum of 27 characters.
    Description: A short, representative cluster name to use for host names and other identifying
names.
    Type: String
  InfrastructureName:
    AllowedPattern: ^([a-zA-Z][a-zA-Z0-9\-]{0,26})$
    MaxLength: 27
    MinLength: 1
    ConstraintDescription: Infrastructure name must be alphanumeric, start with a letter, and have a
maximum of 27 characters.
    Description: A short, unique cluster ID used to tag cloud resources and identify items owned or
used by the cluster.
```

```
    Type: String
  HostedZoneId:
    Description: The Route53 public zone ID to register the targets with, such as
Z21IXYZABCZ2A4.
    Type: String
  HostedZoneName:
    Description: The Route53 zone to register the targets with, such as example.com. Omit the
trailing period.
    Type: String
    Default: "example.com"
  PublicSubnets:
    Description: The internet-facing subnets.
    Type: List<AWS::EC2::Subnet::Id>
  PrivateSubnets:
    Description: The internal subnets.
    Type: List<AWS::EC2::Subnet::Id>
  VpcId:
    Description: The VPC-scoped resources will belong to this VPC.
    Type: AWS::EC2::VPC::Id

Metadata:
  AWS::CloudFormation::Interface:
    ParameterGroups:
    - Label:
        default: "Cluster Information"
      Parameters:
      - ClusterName
      - InfrastructureName
    - Label:
        default: "Network Configuration"
      Parameters:
      - VpcId
      - PublicSubnets
      - PrivateSubnets
    - Label:
        default: "DNS"
      Parameters:
      - HostedZoneName
      - HostedZoneId
    ParameterLabels:
      ClusterName:
        default: "Cluster Name"
      InfrastructureName:
        default: "Infrastructure Name"
      VpcId:
        default: "VPC ID"
      PublicSubnets:
        default: "Public Subnets"
      PrivateSubnets:
        default: "Private Subnets"
      HostedZoneName:
        default: "Public Hosted Zone Name"
      HostedZoneId:
        default: "Public Hosted Zone ID"

Resources:
```

```
ExtApiElb:
  Type: AWS::ElasticLoadBalancingV2::LoadBalancer
  Properties:
    Name: !Join ["-", [!Ref InfrastructureName, "ext"]]
    IpAddressType: ipv4
    Subnets: !Ref PublicSubnets
    Type: network

IntApiElb:
  Type: AWS::ElasticLoadBalancingV2::LoadBalancer
  Properties:
    Name: !Join ["-", [!Ref InfrastructureName, "int"]]
    Scheme: internal
    IpAddressType: ipv4
    Subnets: !Ref PrivateSubnets
    Type: network

IntDns:
  Type: "AWS::Route53::HostedZone"
  Properties:
    HostedZoneConfig:
      Comment: "Managed by CloudFormation"
    Name: !Join [".", [!Ref ClusterName, !Ref HostedZoneName]]
    HostedZoneTags:
    - Key: Name
      Value: !Join ["-", [!Ref InfrastructureName, "int"]]
    - Key: !Join ["", ["kubernetes.io/cluster/", !Ref InfrastructureName]]
      Value: "owned"
    VPCs:
    - VPCId: !Ref VpcId
      VPCRegion: !Ref "AWS::Region"

ExternalApiServerRecord:
  Type: AWS::Route53::RecordSetGroup
  Properties:
    Comment: Alias record for the API server
    HostedZoneId: !Ref HostedZoneId
    RecordSets:
    - Name:
        !Join [
          ".",
          ["api", !Ref ClusterName, !Join ["", [!Ref HostedZoneName, "."]]],
        ]
      Type: A
      AliasTarget:
        HostedZoneId: !GetAtt ExtApiElb.CanonicalHostedZoneID
        DNSName: !GetAtt ExtApiElb.DNSName

InternalApiServerRecord:
  Type: AWS::Route53::RecordSetGroup
  Properties:
    Comment: Alias record for the API server
    HostedZoneId: !Ref IntDns
    RecordSets:
    - Name:
        !Join [
```

```
          ".",
          ["api", !Ref ClusterName, !Join ["", [!Ref HostedZoneName, "."]]],
        ]
      Type: A
      AliasTarget:
        HostedZoneId: !GetAtt IntApiElb.CanonicalHostedZoneID
        DNSName: !GetAtt IntApiElb.DNSName
    - Name:
        !Join [
          ".",
          ["api-int", !Ref ClusterName, !Join ["", [!Ref HostedZoneName, "."]]],
        ]
      Type: A
      AliasTarget:
        HostedZoneId: !GetAtt IntApiElb.CanonicalHostedZoneID
        DNSName: !GetAtt IntApiElb.DNSName

ExternalApiListener:
  Type: AWS::ElasticLoadBalancingV2::Listener
  Properties:
    DefaultActions:
    - Type: forward
      TargetGroupArn:
        Ref: ExternalApiTargetGroup
    LoadBalancerArn:
      Ref: ExtApiElb
    Port: 6443
    Protocol: TCP

ExternalApiTargetGroup:
  Type: AWS::ElasticLoadBalancingV2::TargetGroup
  Properties:
    HealthCheckIntervalSeconds: 10
    HealthCheckPath: "/readyz"
    HealthCheckPort: 6443
    HealthCheckProtocol: HTTPS
    HealthyThresholdCount: 2
    UnhealthyThresholdCount: 2
    Port: 6443
    Protocol: TCP
    TargetType: ip
    VpcId:
      Ref: VpcId
    TargetGroupAttributes:
    - Key: deregistration_delay.timeout_seconds
      Value: 60

InternalApiListener:
  Type: AWS::ElasticLoadBalancingV2::Listener
  Properties:
    DefaultActions:
    - Type: forward
      TargetGroupArn:
        Ref: InternalApiTargetGroup
    LoadBalancerArn:
      Ref: IntApiElb
```

```
    Port: 6443
    Protocol: TCP

  InternalApiTargetGroup:
    Type: AWS::ElasticLoadBalancingV2::TargetGroup
    Properties:
      HealthCheckIntervalSeconds: 10
      HealthCheckPath: "/readyz"
      HealthCheckPort: 6443
      HealthCheckProtocol: HTTPS
      HealthyThresholdCount: 2
      UnhealthyThresholdCount: 2
      Port: 6443
      Protocol: TCP
      TargetType: ip
      VpcId:
        Ref: VpcId
      TargetGroupAttributes:
      - Key: deregistration_delay.timeout_seconds
        Value: 60

  InternalServiceInternalListener:
    Type: AWS::ElasticLoadBalancingV2::Listener
    Properties:
      DefaultActions:
      - Type: forward
        TargetGroupArn:
          Ref: InternalServiceTargetGroup
      LoadBalancerArn:
        Ref: IntApiElb
      Port: 22623
      Protocol: TCP

  InternalServiceTargetGroup:
    Type: AWS::ElasticLoadBalancingV2::TargetGroup
    Properties:
      HealthCheckIntervalSeconds: 10
      HealthCheckPath: "/healthz"
      HealthCheckPort: 22623
      HealthCheckProtocol: HTTPS
      HealthyThresholdCount: 2
      UnhealthyThresholdCount: 2
      Port: 22623
      Protocol: TCP
      TargetType: ip
      VpcId:
        Ref: VpcId
      TargetGroupAttributes:
      - Key: deregistration_delay.timeout_seconds
        Value: 60

  RegisterTargetLambdaIamRole:
    Type: AWS::IAM::Role
    Properties:
      RoleName: !Join ["-", [!Ref InfrastructureName, "nlb", "lambda", "role"]]
      AssumeRolePolicyDocument:
```

```
        Version: "2012-10-17"
        Statement:
        - Effect: "Allow"
          Principal:
            Service:
            - "lambda.amazonaws.com"
          Action:
          - "sts:AssumeRole"
      Path: "/"
      Policies:
      - PolicyName: !Join ["-", [!Ref InfrastructureName, "master", "policy"]]
        PolicyDocument:
          Version: "2012-10-17"
          Statement:
          - Effect: "Allow"
            Action:
              [
                "elasticloadbalancing:RegisterTargets",
                "elasticloadbalancing:DeregisterTargets",
              ]
            Resource: !Ref InternalApiTargetGroup
          - Effect: "Allow"
            Action:
              [
                "elasticloadbalancing:RegisterTargets",
                "elasticloadbalancing:DeregisterTargets",
              ]
            Resource: !Ref InternalServiceTargetGroup
          - Effect: "Allow"
            Action:
              [
                "elasticloadbalancing:RegisterTargets",
                "elasticloadbalancing:DeregisterTargets",
              ]
            Resource: !Ref ExternalApiTargetGroup

  RegisterNlbIpTargets:
    Type: "AWS::Lambda::Function"
    Properties:
      Handler: "index.handler"
      Role:
        Fn::GetAtt:
        - "RegisterTargetLambdaIamRole"
        - "Arn"
      Code:
        ZipFile: |
          import json
          import boto3
          import cfnresponse
          def handler(event, context):
            elb = boto3.client('elbv2')
            if event['RequestType'] == 'Delete':
              elb.deregister_targets(TargetGroupArn=event['ResourceProperties']
['TargetArn'],Targets=[{'Id': event['ResourceProperties']['TargetIp']}])
            elif event['RequestType'] == 'Create':
              elb.register_targets(TargetGroupArn=event['ResourceProperties']['TargetArn'],Targets=
```

```
[{'Id': event['ResourceProperties']['TargetIp']}])
            responseData = {}
            cfnresponse.send(event, context, cfnresponse.SUCCESS, responseData,
event['ResourceProperties']['TargetArn']+event['ResourceProperties']['TargetIp'])
      Runtime: "python3.7"
      Timeout: 120

  RegisterSubnetTagsLambdaIamRole:
    Type: AWS::IAM::Role
    Properties:
      RoleName: !Join ["-", [!Ref InfrastructureName, "subnet-tags-lambda-role"]]
      AssumeRolePolicyDocument:
        Version: "2012-10-17"
        Statement:
        - Effect: "Allow"
          Principal:
            Service:
            - "lambda.amazonaws.com"
          Action:
          - "sts:AssumeRole"
      Path: "/"
      Policies:
      - PolicyName: !Join ["-", [!Ref InfrastructureName, "subnet-tagging-policy"]]
        PolicyDocument:
          Version: "2012-10-17"
          Statement:
          - Effect: "Allow"
            Action:
              [
                "ec2:DeleteTags",
                "ec2:CreateTags"
              ]
            Resource: "arn:aws:ec2:*:*:subnet/*"
          - Effect: "Allow"
            Action:
              [
                "ec2:DescribeSubnets",
                "ec2:DescribeTags"
              ]
            Resource: "*"

  RegisterSubnetTags:
    Type: "AWS::Lambda::Function"
    Properties:
      Handler: "index.handler"
      Role:
        Fn::GetAtt:
        - "RegisterSubnetTagsLambdaIamRole"
        - "Arn"
      Code:
        ZipFile: |
          import json
          import boto3
          import cfnresponse
          def handler(event, context):
            ec2_client = boto3.client('ec2')
```

```
            if event['RequestType'] == 'Delete':
              for subnet_id in event['ResourceProperties']['Subnets']:
                ec2_client.delete_tags(Resources=[subnet_id], Tags=[{'Key': 'kubernetes.io/cluster/' +
event['ResourceProperties']['InfrastructureName']}]);
            elif event['RequestType'] == 'Create':
              for subnet_id in event['ResourceProperties']['Subnets']:
                ec2_client.create_tags(Resources=[subnet_id], Tags=[{'Key': 'kubernetes.io/cluster/' +
event['ResourceProperties']['InfrastructureName'], 'Value': 'shared'}]);
            responseData = {}
            cfnresponse.send(event, context, cfnresponse.SUCCESS, responseData,
event['ResourceProperties']['InfrastructureName']+event['ResourceProperties']['Subnets'][0])
      Runtime: "python3.7"
      Timeout: 120

  RegisterPublicSubnetTags:
    Type: Custom::SubnetRegister
    Properties:
      ServiceToken: !GetAtt RegisterSubnetTags.Arn
      InfrastructureName: !Ref InfrastructureName
      Subnets: !Ref PublicSubnets

  RegisterPrivateSubnetTags:
    Type: Custom::SubnetRegister
    Properties:
      ServiceToken: !GetAtt RegisterSubnetTags.Arn
      InfrastructureName: !Ref InfrastructureName
      Subnets: !Ref PrivateSubnets

Outputs:
  PrivateHostedZoneId:
    Description: Hosted zone ID for the private DNS, which is required for private records.
    Value: !Ref IntDns
  ExternalApiLoadBalancerName:
    Description: Full name of the external API load balancer.
    Value: !GetAtt ExtApiElb.LoadBalancerFullName
  InternalApiLoadBalancerName:
    Description: Full name of the internal API load balancer.
    Value: !GetAtt IntApiElb.LoadBalancerFullName
  ApiServerDnsName:
    Description: Full hostname of the API server, which is required for the Ignition config files.
    Value: !Join [".", ["api-int", !Ref ClusterName, !Ref HostedZoneName]]
  RegisterNlbIpTargetsLambda:
    Description: Lambda ARN useful to help register or deregister IP targets for these load
balancers.
    Value: !GetAtt RegisterNlbIpTargets.Arn
  ExternalApiTargetGroupArn:
    Description: ARN of the external API target group.
    Value: !Ref ExternalApiTargetGroup
  InternalApiTargetGroupArn:
    Description: ARN of the internal API target group.
    Value: !Ref InternalApiTargetGroup
  InternalServiceTargetGroupArn:
    Description: ARN of the internal service target group.
    Value: !Ref InternalServiceTargetGroup
```

重要

如果要将集群部署到 AWS 政府区域，您必须更新 **InternalApiServerRecord** 以使用 **CNAME** 记录。AWS 政府区不支持 **ALIAS** 类型的记录。例如：

```
Type: CNAME
TTL: 10
ResourceRecords:
- !GetAtt IntApiElb.DNSName
```

其他资源

- 有关列出公共托管区的更多信息，请参阅 AWS 文档中的列出公共托管区。

2.11.10. 在 AWS 中创建安全组和角色

您必须在 Amazon Web Services (AWS) 中创建安全组和角色，供您的 OpenShift Container Platform 集群使用。

您可以使用提供的 CloudFormation 模板和自定义参数文件来创建 AWS 资源堆栈。堆栈代表 OpenShift Container Platform 集群所需的安全组和角色。

注意

如果不使用提供的 CloudFormation 模板来创建 AWS 基础架构，您必须检查提供的信息并手动创建基础架构。如果集群没有正确初始化，您可能需要联系红帽支持并提供您的安装日志。

先决条件

- 已配置了一个 AWS 帐户。
- 您可以通过运行 **aws configure**，将 AWS 密钥和区域添加到本地 AWS 配置集中。
- 已为集群生成 Ignition 配置文件。
- 您在 AWS 中创建并配置了 VPC 及相关子网。

流程

1. 创建一个 JSON 文件，其包含模板所需的参数值：

```
[
  {
    "ParameterKey": "InfrastructureName", 1
    "ParameterValue": "mycluster-<random_string>" 2
  },
  {
    "ParameterKey": "VpcCidr", 3
    "ParameterValue": "10.0.0.0/16" 4
  },
  {
    "ParameterKey": "PrivateSubnets", 5
    "ParameterValue": "subnet-<random_string>" 6
```

```
  },
  {
    "ParameterKey": "VpcId", 7
    "ParameterValue": "vpc-<random_string>" 8
  }
]
```

1. 您的 Ignition 配置文件中为集群编码的集群基础架构名称。
2. 指定从 Ignition 配置文件元数据中提取的基础架构名称，其格式为 **<cluster-name>-<random-string>**。
3. VPC 的 CIDR 块。
4. 指定以 **x.x.x.x/16-24** 格式定义的用于 VPC 的 CIDR 地址块。
5. 为 VPC 创建的专用子网。
6. 指定 VPC 的 CloudFormation 模板输出的 **PrivateSubnetIds** 值。
7. 为集群创建的 VPC。
8. 指定 VPC 的 CloudFormation 模板输出的 **VpcId** 值。

2. 复制本主题的**安全对象的 CloudFormation 模板**部分中的模板，并将它以 YAML 文件形式保存到计算机上。此模板描述了集群所需的安全组和角色。
3. 启动 CloudFormation 模板，以创建代表安全组和角色的 AWS 资源堆栈：

重要

您必须在一行内输入命令。

```
$ aws cloudformation create-stack --stack-name <name> 1
    --template-body file://<template>.yaml 2
    --parameters file://<parameters>.json 3
    --capabilities CAPABILITY_NAMED_IAM 4
```

1. **<name>** 是 CloudFormation 堆栈的名称，如 **cluster-sec**。如果您删除集群，则需要此堆栈的名称。
2. **<template>** 是您保存的 CloudFormation 模板 YAML 文件的相对路径和名称。
3. **<parameters>** 是 CloudFormation 参数 JSON 文件的相对路径和名称。
4. 您必须明确声明 **CAPABILITY_NAMED_IAM** 功能，因为提供的模板会创建一些 **AWS::IAM::Role** 和 **AWS::IAM::InstanceProfile** 资源。

输出示例

```
arn:aws:cloudformation:us-east-1:269333783861:stack/cluster-sec/03bd4210-2ed7-11eb-6d7a-13fc0b61e9db
```

4. 确认模板组件已存在：

```
$ aws cloudformation describe-stacks --stack-name <name>
```

在 **StackStatus** 显示 **CREATE_COMPLETE** 后，输出会显示以下参数的值。您必须将这些参数值提供给您在创建集群时要运行的其他 CloudFormation 模板：

MasterSecurityGroupId	Master 安全组 ID
WorkerSecurityGroupId	worker 安全组 ID
MasterInstanceProfile	Master IAM 实例配置集
WorkerInstanceProfile	worker IAM 实例配置集

2.11.10.1. 安全对象的 CloudFormation 模板

您可以使用以下 CloudFormation 模板来部署 OpenShift Container Platform 集群所需的安全对象。

例 2.45. 安全对象的 CloudFormation 模板

```
AWSTemplateFormatVersion: 2010-09-09
Description: Template for OpenShift Cluster Security Elements (Security Groups & IAM)

Parameters:
  InfrastructureName:
    AllowedPattern: ^([a-zA-Z][a-zA-Z0-9\-]{0,26})$
    MaxLength: 27
    MinLength: 1
    ConstraintDescription: Infrastructure name must be alphanumeric, start with a letter, and have a maximum of 27 characters.
    Description: A short, unique cluster ID used to tag cloud resources and identify items owned or used by the cluster.
    Type: String
  VpcCidr:
    AllowedPattern: ^(([0-9]|[1-9][0-9]|1[0-9]{2}|2[0-4][0-9]|25[0-5])\.){3}([0-9]|[1-9][0-9]|1[0-9]{2}|2[0-4][0-9]|25[0-5])(\/(1[6-9]|2[0-4]))$
    ConstraintDescription: CIDR block parameter must be in the form x.x.x.x/16-24.
    Default: 10.0.0.0/16
    Description: CIDR block for VPC.
    Type: String
  VpcId:
    Description: The VPC-scoped resources will belong to this VPC.
    Type: AWS::EC2::VPC::Id
  PrivateSubnets:
```

```
    Description: The internal subnets.
    Type: List<AWS::EC2::Subnet::Id>

Metadata:
  AWS::CloudFormation::Interface:
    ParameterGroups:
    - Label:
        default: "Cluster Information"
      Parameters:
      - InfrastructureName
    - Label:
        default: "Network Configuration"
      Parameters:
      - VpcId
      - VpcCidr
      - PrivateSubnets
    ParameterLabels:
      InfrastructureName:
        default: "Infrastructure Name"
      VpcId:
        default: "VPC ID"
      VpcCidr:
        default: "VPC CIDR"
      PrivateSubnets:
        default: "Private Subnets"

Resources:
  MasterSecurityGroup:
    Type: AWS::EC2::SecurityGroup
    Properties:
      GroupDescription: Cluster Master Security Group
      SecurityGroupIngress:
      - IpProtocol: icmp
        FromPort: 0
        ToPort: 0
        CidrIp: !Ref VpcCidr
      - IpProtocol: tcp
        FromPort: 22
        ToPort: 22
        CidrIp: !Ref VpcCidr
      - IpProtocol: tcp
        ToPort: 6443
        FromPort: 6443
        CidrIp: !Ref VpcCidr
      - IpProtocol: tcp
        FromPort: 22623
        ToPort: 22623
        CidrIp: !Ref VpcCidr
      VpcId: !Ref VpcId

  WorkerSecurityGroup:
    Type: AWS::EC2::SecurityGroup
    Properties:
      GroupDescription: Cluster Worker Security Group
      SecurityGroupIngress:
      - IpProtocol: icmp
```

```
      FromPort: 0
      ToPort: 0
      CidrIp: !Ref VpcCidr
    - IpProtocol: tcp
      FromPort: 22
      ToPort: 22
      CidrIp: !Ref VpcCidr
    VpcId: !Ref VpcId

MasterIngressEtcd:
  Type: AWS::EC2::SecurityGroupIngress
  Properties:
    GroupId: !GetAtt MasterSecurityGroup.GroupId
    SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId
    Description: etcd
    FromPort: 2379
    ToPort: 2380
    IpProtocol: tcp

MasterIngressVxlan:
  Type: AWS::EC2::SecurityGroupIngress
  Properties:
    GroupId: !GetAtt MasterSecurityGroup.GroupId
    SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId
    Description: Vxlan packets
    FromPort: 4789
    ToPort: 4789
    IpProtocol: udp

MasterIngressWorkerVxlan:
  Type: AWS::EC2::SecurityGroupIngress
  Properties:
    GroupId: !GetAtt MasterSecurityGroup.GroupId
    SourceSecurityGroupId: !GetAtt WorkerSecurityGroup.GroupId
    Description: Vxlan packets
    FromPort: 4789
    ToPort: 4789
    IpProtocol: udp

MasterIngressGeneve:
  Type: AWS::EC2::SecurityGroupIngress
  Properties:
    GroupId: !GetAtt MasterSecurityGroup.GroupId
    SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId
    Description: Geneve packets
    FromPort: 6081
    ToPort: 6081
    IpProtocol: udp

MasterIngressWorkerGeneve:
  Type: AWS::EC2::SecurityGroupIngress
  Properties:
    GroupId: !GetAtt MasterSecurityGroup.GroupId
    SourceSecurityGroupId: !GetAtt WorkerSecurityGroup.GroupId
    Description: Geneve packets
    FromPort: 6081
```

```
      ToPort: 6081
      IpProtocol: udp

  MasterIngressInternal:
    Type: AWS::EC2::SecurityGroupIngress
    Properties:
      GroupId: !GetAtt MasterSecurityGroup.GroupId
      SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId
      Description: Internal cluster communication
      FromPort: 9000
      ToPort: 9999
      IpProtocol: tcp

  MasterIngressWorkerInternal:
    Type: AWS::EC2::SecurityGroupIngress
    Properties:
      GroupId: !GetAtt MasterSecurityGroup.GroupId
      SourceSecurityGroupId: !GetAtt WorkerSecurityGroup.GroupId
      Description: Internal cluster communication
      FromPort: 9000
      ToPort: 9999
      IpProtocol: tcp

  MasterIngressInternalUDP:
    Type: AWS::EC2::SecurityGroupIngress
    Properties:
      GroupId: !GetAtt MasterSecurityGroup.GroupId
      SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId
      Description: Internal cluster communication
      FromPort: 9000
      ToPort: 9999
      IpProtocol: udp

  MasterIngressWorkerInternalUDP:
    Type: AWS::EC2::SecurityGroupIngress
    Properties:
      GroupId: !GetAtt MasterSecurityGroup.GroupId
      SourceSecurityGroupId: !GetAtt WorkerSecurityGroup.GroupId
      Description: Internal cluster communication
      FromPort: 9000
      ToPort: 9999
      IpProtocol: udp

  MasterIngressKube:
    Type: AWS::EC2::SecurityGroupIngress
    Properties:
      GroupId: !GetAtt MasterSecurityGroup.GroupId
      SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId
      Description: Kubernetes kubelet, scheduler and controller manager
      FromPort: 10250
      ToPort: 10259
      IpProtocol: tcp

  MasterIngressWorkerKube:
    Type: AWS::EC2::SecurityGroupIngress
    Properties:
```

```
      GroupId: !GetAtt MasterSecurityGroup.GroupId
      SourceSecurityGroupId: !GetAtt WorkerSecurityGroup.GroupId
      Description: Kubernetes kubelet, scheduler and controller manager
      FromPort: 10250
      ToPort: 10259
      IpProtocol: tcp

  MasterIngressIngressServices:
    Type: AWS::EC2::SecurityGroupIngress
    Properties:
      GroupId: !GetAtt MasterSecurityGroup.GroupId
      SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId
      Description: Kubernetes ingress services
      FromPort: 30000
      ToPort: 32767
      IpProtocol: tcp

  MasterIngressWorkerIngressServices:
    Type: AWS::EC2::SecurityGroupIngress
    Properties:
      GroupId: !GetAtt MasterSecurityGroup.GroupId
      SourceSecurityGroupId: !GetAtt WorkerSecurityGroup.GroupId
      Description: Kubernetes ingress services
      FromPort: 30000
      ToPort: 32767
      IpProtocol: tcp

  MasterIngressIngressServicesUDP:
    Type: AWS::EC2::SecurityGroupIngress
    Properties:
      GroupId: !GetAtt MasterSecurityGroup.GroupId
      SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId
      Description: Kubernetes ingress services
      FromPort: 30000
      ToPort: 32767
      IpProtocol: udp

  MasterIngressWorkerIngressServicesUDP:
    Type: AWS::EC2::SecurityGroupIngress
    Properties:
      GroupId: !GetAtt MasterSecurityGroup.GroupId
      SourceSecurityGroupId: !GetAtt WorkerSecurityGroup.GroupId
      Description: Kubernetes ingress services
      FromPort: 30000
      ToPort: 32767
      IpProtocol: udp

  WorkerIngressVxlan:
    Type: AWS::EC2::SecurityGroupIngress
    Properties:
      GroupId: !GetAtt WorkerSecurityGroup.GroupId
      SourceSecurityGroupId: !GetAtt WorkerSecurityGroup.GroupId
      Description: Vxlan packets
      FromPort: 4789
      ToPort: 4789
      IpProtocol: udp
```

```
  WorkerIngressMasterVxlan:
    Type: AWS::EC2::SecurityGroupIngress
    Properties:
      GroupId: !GetAtt WorkerSecurityGroup.GroupId
      SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId
      Description: Vxlan packets
      FromPort: 4789
      ToPort: 4789
      IpProtocol: udp

  WorkerIngressGeneve:
    Type: AWS::EC2::SecurityGroupIngress
    Properties:
      GroupId: !GetAtt WorkerSecurityGroup.GroupId
      SourceSecurityGroupId: !GetAtt WorkerSecurityGroup.GroupId
      Description: Geneve packets
      FromPort: 6081
      ToPort: 6081
      IpProtocol: udp

  WorkerIngressMasterGeneve:
    Type: AWS::EC2::SecurityGroupIngress
    Properties:
      GroupId: !GetAtt WorkerSecurityGroup.GroupId
      SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId
      Description: Geneve packets
      FromPort: 6081
      ToPort: 6081
      IpProtocol: udp

  WorkerIngressInternal:
    Type: AWS::EC2::SecurityGroupIngress
    Properties:
      GroupId: !GetAtt WorkerSecurityGroup.GroupId
      SourceSecurityGroupId: !GetAtt WorkerSecurityGroup.GroupId
      Description: Internal cluster communication
      FromPort: 9000
      ToPort: 9999
      IpProtocol: tcp

  WorkerIngressMasterInternal:
    Type: AWS::EC2::SecurityGroupIngress
    Properties:
      GroupId: !GetAtt WorkerSecurityGroup.GroupId
      SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId
      Description: Internal cluster communication
      FromPort: 9000
      ToPort: 9999
      IpProtocol: tcp

  WorkerIngressInternalUDP:
    Type: AWS::EC2::SecurityGroupIngress
    Properties:
      GroupId: !GetAtt WorkerSecurityGroup.GroupId
      SourceSecurityGroupId: !GetAtt WorkerSecurityGroup.GroupId
```

```
    Description: Internal cluster communication
    FromPort: 9000
    ToPort: 9999
    IpProtocol: udp

  WorkerIngressMasterInternalUDP:
    Type: AWS::EC2::SecurityGroupIngress
    Properties:
      GroupId: !GetAtt WorkerSecurityGroup.GroupId
      SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId
      Description: Internal cluster communication
      FromPort: 9000
      ToPort: 9999
      IpProtocol: udp

  WorkerIngressKube:
    Type: AWS::EC2::SecurityGroupIngress
    Properties:
      GroupId: !GetAtt WorkerSecurityGroup.GroupId
      SourceSecurityGroupId: !GetAtt WorkerSecurityGroup.GroupId
      Description: Kubernetes secure kubelet port
      FromPort: 10250
      ToPort: 10250
      IpProtocol: tcp

  WorkerIngressWorkerKube:
    Type: AWS::EC2::SecurityGroupIngress
    Properties:
      GroupId: !GetAtt WorkerSecurityGroup.GroupId
      SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId
      Description: Internal Kubernetes communication
      FromPort: 10250
      ToPort: 10250
      IpProtocol: tcp

  WorkerIngressIngressServices:
    Type: AWS::EC2::SecurityGroupIngress
    Properties:
      GroupId: !GetAtt WorkerSecurityGroup.GroupId
      SourceSecurityGroupId: !GetAtt WorkerSecurityGroup.GroupId
      Description: Kubernetes ingress services
      FromPort: 30000
      ToPort: 32767
      IpProtocol: tcp

  WorkerIngressMasterIngressServices:
    Type: AWS::EC2::SecurityGroupIngress
    Properties:
      GroupId: !GetAtt WorkerSecurityGroup.GroupId
      SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId
      Description: Kubernetes ingress services
      FromPort: 30000
      ToPort: 32767
      IpProtocol: tcp

  WorkerIngressIngressServicesUDP:
```

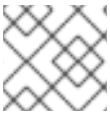
```
  Type: AWS::EC2::SecurityGroupIngress
  Properties:
    GroupId: !GetAtt WorkerSecurityGroup.GroupId
    SourceSecurityGroupId: !GetAtt WorkerSecurityGroup.GroupId
    Description: Kubernetes ingress services
    FromPort: 30000
    ToPort: 32767
    IpProtocol: udp

WorkerIngressMasterIngressServicesUDP:
  Type: AWS::EC2::SecurityGroupIngress
  Properties:
    GroupId: !GetAtt WorkerSecurityGroup.GroupId
    SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId
    Description: Kubernetes ingress services
    FromPort: 30000
    ToPort: 32767
    IpProtocol: udp

MasterIamRole:
  Type: AWS::IAM::Role
  Properties:
    AssumeRolePolicyDocument:
      Version: "2012-10-17"
      Statement:
      - Effect: "Allow"
        Principal:
          Service:
          - "ec2.amazonaws.com"
        Action:
        - "sts:AssumeRole"
    Policies:
    - PolicyName: !Join ["-", [!Ref InfrastructureName, "master", "policy"]]
      PolicyDocument:
        Version: "2012-10-17"
        Statement:
        - Effect: "Allow"
          Action:
          - "ec2:AttachVolume"
          - "ec2:AuthorizeSecurityGroupIngress"
          - "ec2:CreateSecurityGroup"
          - "ec2:CreateTags"
          - "ec2:CreateVolume"
          - "ec2:DeleteSecurityGroup"
          - "ec2:DeleteVolume"
          - "ec2:Describe*"
          - "ec2:DetachVolume"
          - "ec2:ModifyInstanceAttribute"
          - "ec2:ModifyVolume"
          - "ec2:RevokeSecurityGroupIngress"
          - "elasticloadbalancing:AddTags"
          - "elasticloadbalancing:AttachLoadBalancerToSubnets"
          - "elasticloadbalancing:ApplySecurityGroupsToLoadBalancer"
          - "elasticloadbalancing:CreateListener"
          - "elasticloadbalancing:CreateLoadBalancer"
          - "elasticloadbalancing:CreateLoadBalancerPolicy"
```

```
          - "elasticloadbalancing:CreateLoadBalancerListeners"
          - "elasticloadbalancing:CreateTargetGroup"
          - "elasticloadbalancing:ConfigureHealthCheck"
          - "elasticloadbalancing:DeleteListener"
          - "elasticloadbalancing:DeleteLoadBalancer"
          - "elasticloadbalancing:DeleteLoadBalancerListeners"
          - "elasticloadbalancing:DeleteTargetGroup"
          - "elasticloadbalancing:DeregisterInstancesFromLoadBalancer"
          - "elasticloadbalancing:DeregisterTargets"
          - "elasticloadbalancing:Describe*"
          - "elasticloadbalancing:DetachLoadBalancerFromSubnets"
          - "elasticloadbalancing:ModifyListener"
          - "elasticloadbalancing:ModifyLoadBalancerAttributes"
          - "elasticloadbalancing:ModifyTargetGroup"
          - "elasticloadbalancing:ModifyTargetGroupAttributes"
          - "elasticloadbalancing:RegisterInstancesWithLoadBalancer"
          - "elasticloadbalancing:RegisterTargets"
          - "elasticloadbalancing:SetLoadBalancerPoliciesForBackendServer"
          - "elasticloadbalancing:SetLoadBalancerPoliciesOfListener"
          - "kms:DescribeKey"
          Resource: "*"

MasterInstanceProfile:
  Type: "AWS::IAM::InstanceProfile"
  Properties:
    Roles:
    - Ref: "MasterIamRole"

WorkerIamRole:
  Type: AWS::IAM::Role
  Properties:
    AssumeRolePolicyDocument:
      Version: "2012-10-17"
      Statement:
      - Effect: "Allow"
        Principal:
          Service:
          - "ec2.amazonaws.com"
        Action:
        - "sts:AssumeRole"
    Policies:
    - PolicyName: !Join ["-", [!Ref InfrastructureName, "worker", "policy"]]
      PolicyDocument:
        Version: "2012-10-17"
        Statement:
        - Effect: "Allow"
          Action:
          - "ec2:DescribeInstances"
          - "ec2:DescribeRegions"
          Resource: "*"

WorkerInstanceProfile:
  Type: "AWS::IAM::InstanceProfile"
  Properties:
    Roles:
    - Ref: "WorkerIamRole"
```

```
Outputs:
  MasterSecurityGroupId:
    Description: Master Security Group ID
    Value: !GetAtt MasterSecurityGroup.GroupId

  WorkerSecurityGroupId:
    Description: Worker Security Group ID
    Value: !GetAtt WorkerSecurityGroup.GroupId

  MasterInstanceProfile:
    Description: Master IAM Instance Profile
    Value: !Ref MasterInstanceProfile

  WorkerInstanceProfile:
    Description: Worker IAM Instance Profile
    Value: !Ref WorkerInstanceProfile
```

2.11.11. AWS 基础架构的 RHCOS AMI

红帽为您为 OpenShift Container Platform 节点指定的各种 Amazon Web Services（AWS）区域提供了有效的 Red Hat Enterprise Linux CoreOS（RHCOS）AMI。



注意

您还可以导入您自己的 AMI，来安装到没有发布 RHCOS AMI 的区域。

表 2.33. RHCOS AMI

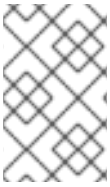
AWS 区	AWS AMI
af-south-1	**ami-09921c9c1c36e695c**
ap-east-1	**ami-01ee8446e9af6b197**
ap-northeast-1	**ami-04e5b5722a55846ea**
ap-northeast-2	**ami-0fdc25c8a0273a742**
ap-south-1	**ami-09e3deb397cc526a8**
ap-southeast-1	**ami-0630e03f75e02eec4**
ap-southeast-2	**ami-069450613262ba03c**
ca-central-1	**ami-012518cdbd3057dfd**
eu-central-1	**ami-0bd7175ff5b1aef0c**

AWS 区	AWS AMI
eu-north-1	**ami-06c9ec42d0a839ad2**
eu-south-1	**ami-0614d7440a0363d71**
eu-west-1	**ami-01b89df58b5d4d5fa**
eu-west-2	**ami-06f6e31ddd554f89d**
eu-west-3	**ami-0dc82e2517ded15a1**
me-south-1	**ami-07d181e3aa0f76067**
sa-east-1	**ami-0cd44e6dd20e6c7fa**
us-east-1	**ami-04a16d506e5b0e246**
us-east-2	**ami-0a1f868ad58ea59a7**
us-west-1	**ami-0a65d76e3a6f6622f**
us-west-2	**ami-0dd9008abadc519f1**

2.11.12. 在 AWS 中创建 bootstrap 节点

您必须在 Amazon Web Services (AWS) 中创建 bootstrap 节点，以便在 OpenShift Container Platform 集群初始化过程中使用。

您可以使用提供的 CloudFormation 模板和自定义参数文件来创建 AWS 资源堆栈。堆栈代表 OpenShift Container Platform 安装所需的 bootstrap 节点。



注意

如果不使用提供的 CloudFormation 模板来创建 bootstrap 节点，您必须检查提供的信息并手动创建基础架构。如果集群没有正确初始化，您可能需要联系红帽支持并提供您的安装日志。

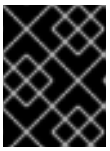
先决条件

- 已配置了一个 AWS 帐户。
- 您可以通过运行 **aws configure**，将 AWS 密钥和区域添加到本地 AWS 配置集中。
- 已为集群生成 Ignition 配置文件。
- 您在 AWS 中创建并配置了 VPC 及相关子网。
- 您在 AWS 中创建并配置了 DNS、负载均衡器和监听程序。

- 您在 AWS 中创建了集群所需的安全组和角色。

流程

1. 提供一个位置，以便向集群提供 **bootstrap.ign** Ignition 配置文件。此文件位于您的安装目录中。达成此目标的一种方式是在集群区域中创建一个 S3 存储桶，并将 Ignition 配置文件上传到其中。



重要

提供的 CloudFormation 模板假定集群的 Ignition 配置文件由 S3 存储桶提供。如果选择从其他位置提供文件，您必须修改模板。

重要

如果您部署到具有与 AWS SDK 不同的端点，或者您提供自己的自定义端点的区域，则必须为 S3 存储桶使用预签名 URL 而不是 **s3://** 模式。

注意

bootstrap Ignition 配置文件包含 secret，如 X.509 密钥。以下步骤为 S3 存储桶提供基本安全性。若要提供额外的安全性，您可以启用 S3 存储桶策略，仅允许某些用户（如 OpenShift IAM 用户）访问存储桶中包含的对象。您可以完全避开 S3，并从 bootstrap 可访问的任意地址提供 bootstrap Ignition 配置文件。

a. 创建存储桶：

```
$ aws s3 mb s3://<cluster-name>-infra 1
```

1 **<cluster-name>-infra** 是存储桶名称。在创建 **install-config.yaml** 文件时，将 **<cluster-name>** 替换为为集群指定的名称。

b. 将 **bootstrap.ign** Ignition 配置文件上传到存储桶：

```
$ aws s3 cp <installation_directory>/bootstrap.ign s3://<cluster-name>-
infra/bootstrap.ign 1
```

1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

c. 验证文件已经上传：

```
$ aws s3 ls s3://<cluster-name>-infra/
```

输出示例

```
2019-04-03 16:15:16     314878 bootstrap.ign
```

2. 创建一个 JSON 文件，其包含模板所需的参数值：

```
[
  {
    "ParameterKey": "InfrastructureName", 1
```

```
    "ParameterValue": "mycluster-<random_string>" 2
  },
  {
    "ParameterKey": "RhcosAmi", 3
    "ParameterValue": "ami-<random_string>" 4
  },
  {
    "ParameterKey": "AllowedBootstrapSshCidr", 5
    "ParameterValue": "0.0.0.0/0" 6
  },
  {
    "ParameterKey": "PublicSubnet", 7
    "ParameterValue": "subnet-<random_string>" 8
  },
  {
    "ParameterKey": "MasterSecurityGroupId", 9
    "ParameterValue": "sg-<random_string>" 10
  },
  {
    "ParameterKey": "VpcId", 11
    "ParameterValue": "vpc-<random_string>" 12
  },
  {
    "ParameterKey": "BootstrapIgnitionLocation", 13
    "ParameterValue": "s3://<bucket_name>/bootstrap.ign" 14
  },
  {
    "ParameterKey": "AutoRegisterELB", 15
    "ParameterValue": "yes" 16
  },
  {
    "ParameterKey": "RegisterNlbIpTargetsLambdaArn", 17
    "ParameterValue": "arn:aws:lambda:<region>:<account_number>:function:
<dns_stack_name>-RegisterNlbIpTargets-<random_string>" 18
  },
  {
    "ParameterKey": "ExternalApiTargetGroupArn", 19
    "ParameterValue": "arn:aws:elasticloadbalancing:<region>:
<account_number>:targetgroup/<dns_stack_name>-Exter-<random_string>" 20
  },
  {
    "ParameterKey": "InternalApiTargetGroupArn", 21
    "ParameterValue": "arn:aws:elasticloadbalancing:<region>:
<account_number>:targetgroup/<dns_stack_name>-Inter-<random_string>" 22
  },
  {
    "ParameterKey": "InternalServiceTargetGroupArn", 23
    "ParameterValue": "arn:aws:elasticloadbalancing:<region>:
<account_number>:targetgroup/<dns_stack_name>-Inter-<random_string>" 24
  }
]
```

1 您的 Ignition 配置文件中为集群编码的集群基础架构名称。

2. 指定从 Ignition 配置文件元数据中提取的基础架构名称，其格式为 **<cluster-name>-<random-string>**。
3. 用于 bootstrap 节点的当前 Red Hat Enterprise Linux CoreOS (RHCOS) AMI。
4. 指定有效的 **AWS::EC2::Image::Id** 值。
5. 允许通过 SSH 访问 bootstrap 节点的 CIDR 块。
6. 以 **x.x.x.x/16-24** 格式指定 CIDR 块。
7. 与 VPC 关联的公共子网，将 bootstrap 节点启动到其中。
8. 指定 VPC 的 CloudFormation 模板输出的 **PublicSubnetIds** 值。
9. master 安全组 ID（用于注册临时规则）
10. 指定安全组和角色的 CloudFormation 模板输出的 **MasterSecurityGroupId** 值。
11. 创建的资源将从属于的 VPC。
12. 指定 VPC 的 CloudFormation 模板输出的 **VpcId** 值。
13. 从中获取 bootstrap Ignition 配置文件的位置。
14. 指定 S3 存储桶和文件名，格式为 **s3://<bucket_name>/bootstrap.ign**。
15. 是否要注册网络负载均衡器 (NLB) 。
16. 指定 **yes** 或 **no**。如果指定 **yes**，您必须提供一个 Lambda Amazon Resource Name (ARN) 值。
17. NLB IP 目标注册 lambda 组的 ARN。
18. 指定 DNS 和负载均衡的 CloudFormation 模板输出的 **RegisterNlbIpTargetsLambda** 值。如果将集群部署到 AWS GovCloud 区域，请使用 **arn:aws-us-gov**。
19. 外部 API 负载均衡器目标组的 ARN。
20. 指定 DNS 和负载均衡的 CloudFormation 模板输出的 **ExternalApiTargetGroupArn** 值。如果将集群部署到 AWS GovCloud 区域，请使用 **arn:aws-us-gov**。
21. 内部 API 负载均衡器目标组群的 ARN。
22. 指定 DNS 和负载均衡的 CloudFormation 模板输出的 **InternalApiTargetGroupArn** 值。如果将集群部署到 AWS GovCloud 区域，请使用 **arn:aws-us-gov**。
23. 内部服务负载均衡器目标组群的 ARN。
24. 指定 DNS 和负载均衡的 CloudFormation 模板输出的 **InternalServiceTargetGroupArn** 值。如果将集群部署到 AWS GovCloud 区域，请使用 **arn:aws-us-gov**。

3. 复制本主题的 **Bootstrap 机器的 CloudFormation 模板**部分中的模板，并将它以 YAML 文件形式保存到计算机上。此模板描述了集群所需的 bootstrap 机器。
4. 启动 CloudFormation 模板，以创建代表 bootstrap 节点的 AWS 资源堆栈：

重要

您必须在一行内输入命令。

```
$ aws cloudformation create-stack --stack-name <name> 1
    --template-body file://<template>.yaml 2
    --parameters file://<parameters>.json 3
    --capabilities CAPABILITY_NAMED_IAM 4
```

1. **<name>** 是 CloudFormation 堆栈的名称，如 **cluster-bootstrap**。如果您删除集群，则需要此堆栈的名称。
2. **<template>** 是您保存的 CloudFormation 模板 YAML 文件的相对路径和名称。
3. **<parameters>** 是 CloudFormation 参数 JSON 文件的相对路径和名称。
4. 您必须明确声明 **CAPABILITY_NAMED_IAM** 功能，因为提供的模板会创建一些 **AWS::IAM::Role** 和 **AWS::IAM::InstanceProfile** 资源。

输出示例

```
arn:aws:cloudformation:us-east-1:269333783861:stack/cluster-bootstrap/12944486-2add-11eb-9dee-12dace8e3a83
```

5. 确认模板组件已存在：

```
$ aws cloudformation describe-stacks --stack-name <name>
```

在 **StackStatus** 显示 **CREATE_COMPLETE** 后，输出会显示以下参数的值。您必须将这些参数值提供给您在创建集群时要运行的其他 CloudFormation 模板：

Bootstrap InstanceId	bootstrap 实例 ID。
Bootstrap PublicIp	bootstrap 节点公共 IP 地址。
Bootstrap PrivateIp	bootstrap 节点专用 IP 地址。

2.11.12.1. bootstrap 机器的 CloudFormation 模板

您可以使用以下 CloudFormation 模板来部署 OpenShift Container Platform 集群所需的 bootstrap 机器。

例 2.46. bootstrap 机器的 CloudFormation 模板

```
AWSTemplateFormatVersion: 2010-09-09
Description: Template for OpenShift Cluster Bootstrap (EC2 Instance, Security Groups and IAM)
```

```
Parameters:
  InfrastructureName:
    AllowedPattern: ^([a-zA-Z][a-zA-Z0-9\-]{0,26})$
    MaxLength: 27
    MinLength: 1
    ConstraintDescription: Infrastructure name must be alphanumeric, start with a letter, and have a
maximum of 27 characters.
    Description: A short, unique cluster ID used to tag cloud resources and identify items owned or
used by the cluster.
    Type: String
  RhcosAmi:
    Description: Current Red Hat Enterprise Linux CoreOS AMI to use for bootstrap.
    Type: AWS::EC2::Image::Id
  AllowedBootstrapSshCidr:
    AllowedPattern: ^(([0-9]|[1-9][0-9]|1[0-9]{2}|2[0-4][0-9]|25[0-5])\.){3}([0-9]|[1-9][0-9]|1[0-9]{2}|2[0-
4][0-9]|25[0-5])(\/([0-9]|1[0-9]|2[0-9]|3[0-2]))$
    ConstraintDescription: CIDR block parameter must be in the form x.x.x.x/0-32.
    Default: 0.0.0.0/0
    Description: CIDR block to allow SSH access to the bootstrap node.
    Type: String
  PublicSubnet:
    Description: The public subnet to launch the bootstrap node into.
    Type: AWS::EC2::Subnet::Id
  MasterSecurityGroupId:
    Description: The master security group ID for registering temporary rules.
    Type: AWS::EC2::SecurityGroup::Id
  VpcId:
    Description: The VPC-scoped resources will belong to this VPC.
    Type: AWS::EC2::VPC::Id
  BootstrapIgnitionLocation:
    Default: s3://my-s3-bucket/bootstrap.ign
    Description: Ignition config file location.
    Type: String
  AutoRegisterELB:
    Default: "yes"
    AllowedValues:
    - "yes"
    - "no"
    Description: Do you want to invoke NLB registration, which requires a Lambda ARN parameter?
    Type: String
  RegisterNlbIpTargetsLambdaArn:
    Description: ARN for NLB IP target registration lambda.
    Type: String
  ExternalApiTargetGroupArn:
    Description: ARN for external API load balancer target group.
    Type: String
  InternalApiTargetGroupArn:
    Description: ARN for internal API load balancer target group.
    Type: String
  InternalServiceTargetGroupArn:
    Description: ARN for internal service load balancer target group.
    Type: String

Metadata:
  AWS::CloudFormation::Interface:
    ParameterGroups:
```

```
    - Label:
        default: "Cluster Information"
      Parameters:
      - InfrastructureName
    - Label:
        default: "Host Information"
      Parameters:
      - RhcosAmi
      - BootstrapIgnitionLocation
      - MasterSecurityGroupId
    - Label:
        default: "Network Configuration"
      Parameters:
      - VpcId
      - AllowedBootstrapSshCidr
      - PublicSubnet
    - Label:
        default: "Load Balancer Automation"
      Parameters:
      - AutoRegisterELB
      - RegisterNlbIpTargetsLambdaArn
      - ExternalApiTargetGroupArn
      - InternalApiTargetGroupArn
      - InternalServiceTargetGroupArn
    ParameterLabels:
      InfrastructureName:
        default: "Infrastructure Name"
      VpcId:
        default: "VPC ID"
      AllowedBootstrapSshCidr:
        default: "Allowed SSH Source"
      PublicSubnet:
        default: "Public Subnet"
      RhcosAmi:
        default: "Red Hat Enterprise Linux CoreOS AMI ID"
      BootstrapIgnitionLocation:
        default: "Bootstrap Ignition Source"
      MasterSecurityGroupId:
        default: "Master Security Group ID"
      AutoRegisterELB:
        default: "Use Provided ELB Automation"

Conditions:
  DoRegistration: !Equals ["yes", !Ref AutoRegisterELB]

Resources:
  BootstrapIamRole:
    Type: AWS::IAM::Role
    Properties:
      AssumeRolePolicyDocument:
        Version: "2012-10-17"
        Statement:
        - Effect: "Allow"
          Principal:
            Service:
            - "ec2.amazonaws.com"
```

```
        Action:
        - "sts:AssumeRole"
      Path: "/"
      Policies:
      - PolicyName: !Join ["-", [!Ref InfrastructureName, "bootstrap", "policy"]]
        PolicyDocument:
          Version: "2012-10-17"
          Statement:
          - Effect: "Allow"
            Action: "ec2:Describe*"
            Resource: "*"
          - Effect: "Allow"
            Action: "ec2:AttachVolume"
            Resource: "*"
          - Effect: "Allow"
            Action: "ec2:DetachVolume"
            Resource: "*"
          - Effect: "Allow"
            Action: "s3:GetObject"
            Resource: "*"

  BootstrapInstanceProfile:
    Type: "AWS::IAM::InstanceProfile"
    Properties:
      Path: "/"
      Roles:
      - Ref: "BootstrapIamRole"

  BootstrapSecurityGroup:
    Type: AWS::EC2::SecurityGroup
    Properties:
      GroupDescription: Cluster Bootstrap Security Group
      SecurityGroupIngress:
      - IpProtocol: tcp
        FromPort: 22
        ToPort: 22
        CidrIp: !Ref AllowedBootstrapSshCidr
      - IpProtocol: tcp
        ToPort: 19531
        FromPort: 19531
        CidrIp: 0.0.0.0/0
      VpcId: !Ref VpcId

  BootstrapInstance:
    Type: AWS::EC2::Instance
    Properties:
      ImageId: !Ref RhcosAmi
      IamInstanceProfile: !Ref BootstrapInstanceProfile
      InstanceType: "i3.large"
      NetworkInterfaces:
      - AssociatePublicIpAddress: "true"
        DeviceIndex: "0"
        GroupSet:
        - !Ref "BootstrapSecurityGroup"
        - !Ref "MasterSecurityGroupId"
        SubnetId: !Ref "PublicSubnet"
```

```
      UserData:
        Fn::Base64: !Sub
        - '{"ignition":{"config":{"replace":{"source":"${S3Loc}"}},"version":"3.1.0"}}'
        - {
          S3Loc: !Ref BootstrapIgnitionLocation
        }

    RegisterBootstrapApiTarget:
      Condition: DoRegistration
      Type: Custom::NLBRegister
      Properties:
        ServiceToken: !Ref RegisterNlbIpTargetsLambdaArn
        TargetArn: !Ref ExternalApiTargetGroupArn
        TargetIp: !GetAtt BootstrapInstance.PrivateIp

    RegisterBootstrapInternalApiTarget:
      Condition: DoRegistration
      Type: Custom::NLBRegister
      Properties:
        ServiceToken: !Ref RegisterNlbIpTargetsLambdaArn
        TargetArn: !Ref InternalApiTargetGroupArn
        TargetIp: !GetAtt BootstrapInstance.PrivateIp

    RegisterBootstrapInternalServiceTarget:
      Condition: DoRegistration
      Type: Custom::NLBRegister
      Properties:
        ServiceToken: !Ref RegisterNlbIpTargetsLambdaArn
        TargetArn: !Ref InternalServiceTargetGroupArn
        TargetIp: !GetAtt BootstrapInstance.PrivateIp

  Outputs:
    BootstrapInstanceId:
      Description: Bootstrap Instance ID.
      Value: !Ref BootstrapInstance

    BootstrapPublicIp:
      Description: The bootstrap node public IP address.
      Value: !GetAtt BootstrapInstance.PublicIp

    BootstrapPrivateIp:
      Description: The bootstrap node private IP address.
      Value: !GetAtt BootstrapInstance.PrivateIp
```

其他资源

- 如需有关 AWS 区的 Red Hat Enterprise Linux CoreOS（RHCOS）AMI 的详细信息，请参阅 AWS 基础架构的 RHCOS AMI。

2.11.13. 在 AWS 中创建 control plane 机器

您必须在集群要使用的 Amazon Web Services（AWS）中创建 control plane 机器。

您可以使用提供的 CloudFormation 模板和自定义参数文件，创建代表 control plane 节点的 AWS 资源堆栈。

重要

CloudFormation 模板会创建一个堆栈，它代表三个 control plane 节点。

注意

如果不使用提供的 CloudFormation 模板来创建 control plane 节点，您必须检查提供的信息并手动创建基础架构。如果集群没有正确初始化，您可能需要联系红帽支持并提供您的安装日志。

先决条件

- 已配置了一个 AWS 帐户。
- 您可以通过运行 **aws configure**，将 AWS 密钥和区域添加到本地 AWS 配置集中。
- 已为集群生成 Ignition 配置文件。
- 您在 AWS 中创建并配置了 VPC 及相关子网。
- 您在 AWS 中创建并配置了 DNS、负载均衡器和监听程序。
- 您在 AWS 中创建了集群所需的安全组和角色。
- 已创建 bootstrap 机器。

流程

1. 创建一个 JSON 文件，其包含模板所需的参数值：

```
[
  {
    "ParameterKey": "InfrastructureName", 1
    "ParameterValue": "mycluster-<random_string>" 2
  },
  {
    "ParameterKey": "RhcosAmi", 3
    "ParameterValue": "ami-<random_string>" 4
  },
  {
    "ParameterKey": "AutoRegisterDNS", 5
    "ParameterValue": "yes" 6
  },
  {
    "ParameterKey": "PrivateHostedZoneId", 7
    "ParameterValue": "<random_string>" 8
  },
  {
    "ParameterKey": "PrivateHostedZoneName", 9
    "ParameterValue": "mycluster.example.com" 10
  },
```

```
  {
    "ParameterKey": "Master0Subnet", (11)
    "ParameterValue": "subnet-<random_string>" (12)
  },
  {
    "ParameterKey": "Master1Subnet", (13)
    "ParameterValue": "subnet-<random_string>" (14)
  },
  {
    "ParameterKey": "Master2Subnet", (15)
    "ParameterValue": "subnet-<random_string>" (16)
  },
  {
    "ParameterKey": "MasterSecurityGroupId", (17)
    "ParameterValue": "sg-<random_string>" (18)
  },
  {
    "ParameterKey": "IgnitionLocation", (19)
    "ParameterValue": "https://api-int.<cluster_name>.<domain_name>:22623/config/master"
(20)
  },
  {
    "ParameterKey": "CertificateAuthorities", (21)
    "ParameterValue": "data:text/plain;charset=utf-8;base64,ABC...xYz==" (22)
  },
  {
    "ParameterKey": "MasterInstanceProfileName", (23)
    "ParameterValue": "<roles_stack>-MasterInstanceProfile-<random_string>" (24)
  },
  {
    "ParameterKey": "MasterInstanceType", (25)
    "ParameterValue": "m4.xlarge" (26)
  },
  {
    "ParameterKey": "AutoRegisterELB", (27)
    "ParameterValue": "yes" (28)
  },
  {
    "ParameterKey": "RegisterNlbIpTargetsLambdaArn", (29)
    "ParameterValue": "arn:aws:lambda:<region>:<account_number>:function:
<dns_stack_name>-RegisterNlbIpTargets-<random_string>" (30)
  },
  {
    "ParameterKey": "ExternalApiTargetGroupArn", (31)
    "ParameterValue": "arn:aws:elasticloadbalancing:<region>:
<account_number>:targetgroup/<dns_stack_name>-Exter-<random_string>" (32)
  },
  {
    "ParameterKey": "InternalApiTargetGroupArn", (33)
    "ParameterValue": "arn:aws:elasticloadbalancing:<region>:
<account_number>:targetgroup/<dns_stack_name>-Inter-<random_string>" (34)
  },
  {
```

```
    "ParameterKey": "InternalServiceTargetGroupArn", 35
    "ParameterValue": "arn:aws:elasticloadbalancing:<region>:
<account_number>:targetgroup/<dns_stack_name>-Inter-<random_string>" 36
  }
]
```

1. 您的 Ignition 配置文件中为集群编码的集群基础架构名称。
2. 指定从 Ignition 配置文件元数据中提取的基础架构名称，其格式为 **<cluster-name>-<random-string>**。
3. 用于 control plane 机器的当前 Red Hat Enterprise Linux CoreOS (RHCOS) AMI。
4. 指定 **AWS::EC2::Image::Id** 值。
5. 是否要执行 DNS etcd 注册。
6. 指定 **yes** 或 **no**。如果指定 **yes**，您必须提供托管区信息。
7. 用来注册 etcd 目标的 Route 53 专用区 ID。
8. 指定 DNS 和负载均衡的 CloudFormation 模板输出的 **PrivateHostedZoneId** 值。
9. 用来注册目标的 Route 53 区。
10. 指定 **<cluster_name>.<domain_name>**，其中 **<domain_name>** 是您为集群生成 **install-config.yaml** 文件时所用的 Route 53 基域。请勿包含 AWS 控制台中显示的结尾句点 (.)。
11. 13. 15. 在其中启动 control plane 机器的子网，最好是专用子网。
12. 14. 16. 从 DNS 和负载均衡的 CloudFormation 模板输出的 **PrivateSubnets** 值指定子网。
17. 与 control plane 节点（也称为 master 节点）关联的 master 安全组 ID。
18. 指定安全组和角色的 CloudFormation 模板输出的 **MasterSecurityGroupId** 值。
19. 从中获取 control plane Ignition 配置文件的位置。
20. 指定生成的 Ignition 配置文件的位置，**https://api-int.<cluster_name>.<domain_name>:22623/config/master**。
21. 要使用的 base64 编码证书颁发机构字符串。
22. 指定安装目录中 **master.ign** 文件中的值。这个值是一个长字符串，格式为 **data:text/plain;charset=utf-8;base64,ABC...xYz==**。
23. 与 control plane 节点关联的 IAM 配置集。
24. 指定安全组和角色的 CloudFormation 模板输出的 **MasterInstanceProfile** 参数值。
25. 用于 control plane 机器的 AWS 实例类型。
26. 允许的值：
    - **m4.xlarge**
    - **m4.2xlarge**

- **m4.4xlarge**
- **m4.8xlarge**
- **m4.10xlarge**
- **m4.16xlarge**
- **m5.xlarge**
- **m5.2xlarge**
- **m5.4xlarge**
- **m5.8xlarge**
- **m5.10xlarge**
- **m5.16xlarge**
- **m6i.xlarge**
- **c4.2xlarge**
- **c4.4xlarge**
- **c4.8xlarge**
- **r4.xlarge**
- **r4.2xlarge**
- **r4.4xlarge**
- **r4.8xlarge**
- **r4.16xlarge**

重要

如果您的区域中没有 **m4** 实例类型，例如 **eu-west-3**，请改为指定 **m5** 类型，如 **m5.xlarge**。

27 是否要注册网络负载均衡器 (NLB) 。

28 指定 **yes** 或 **no**。如果指定 **yes**，您必须提供一个 Lambda Amazon Resource Name (ARN) 值。

29 NLB IP 目标注册 lambda 组的 ARN。

30 指定 DNS 和负载均衡的 CloudFormation 模板输出的 **RegisterNlbIpTargetsLambda** 值。如果将集群部署到 AWS GovCloud 区域，请使用 **arn:aws-us-gov**。

31 外部 API 负载均衡器目标组的 ARN。

32 指定 DNS 和负载均衡的 CloudFormation 模板输出的 **ExternalApiTargetGroupArn** 值。如果将集群部署到 AWS GovCloud 区域，请使用 **arn:aws-us-gov**。

33 内部 API 负载均衡器目标组群的 ARN。

34 指定 DNS 和负载均衡的 CloudFormation 模板输出的 **InternalApiTargetGroupArn** 值。如果将集群部署到 AWS GovCloud 区域，请使用 **arn:aws-us-gov**。

35 内部服务负载均衡器目标组群的 ARN。

36 指定 DNS 和负载均衡的 CloudFormation 模板输出的 **InternalServiceTargetGroupArn** 值。如果将集群部署到 AWS GovCloud 区域，请使用 **arn:aws-us-gov**。

2. 复制**control plane 机器的 CloudFormation 模板**一节中的模板，并将它以 YAML 文件形式保存到计算机上。此模板描述了集群所需的 control plane 机器。

3. 如果您将 **m5** 实例类型指定为 **MasterInstanceType** 的值，请将该实例类型添加到 CloudFormation 模板中的 **MasterInstanceType.AllowedValues** 参数。

4. 启动 CloudFormation 模板，以创建代表 control plane 节点的 AWS 资源堆栈：

重要

您必须在一行内输入命令。

```
$ aws cloudformation create-stack --stack-name <name> 1
    --template-body file://<template>.yaml 2
    --parameters file://<parameters>.json 3
```

1 **<name>** 是 CloudFormation 堆栈的名称，如 **cluster-control-plane**。如果您删除集群，则需要此堆栈的名称。

2 **<template>** 是您保存的 CloudFormation 模板 YAML 文件的相对路径和名称。

3 **<parameters>** 是 CloudFormation 参数 JSON 文件的相对路径和名称。

输出示例

```
arn:aws:cloudformation:us-east-1:269333783861:stack/cluster-control-plane/21c7e2b0-2ee2-11eb-c6f6-0aa34627df4b
```

注意

CloudFormation 模板会创建一个堆栈，它代表三个 control plane 节点。

5. 确认模板组件已存在：

```
$ aws cloudformation describe-stacks --stack-name <name>
```

2.11.13.1. control plane 机器的 CloudFormation 模板

您可以使用以下 CloudFormation 模板来部署 OpenShift Container Platform 集群所需的 control plane 机器。

例 2.47. control plane 机器的 CloudFormation 模板

```
AWSTemplateFormatVersion: 2010-09-09
Description: Template for OpenShift Cluster Node Launch (EC2 master instances)

Parameters:
  InfrastructureName:
    AllowedPattern: ^([a-zA-Z][a-zA-Z0-9\-]{0,26})$
    MaxLength: 27
    MinLength: 1
    ConstraintDescription: Infrastructure name must be alphanumeric, start with a letter, and have a
maximum of 27 characters.
    Description: A short, unique cluster ID used to tag nodes for the kubelet cloud provider.
    Type: String
  RhcosAmi:
    Description: Current Red Hat Enterprise Linux CoreOS AMI to use for bootstrap.
    Type: AWS::EC2::Image::Id
  AutoRegisterDNS:
    Default: "yes"
    AllowedValues:
    - "yes"
    - "no"
    Description: Do you want to invoke DNS etcd registration, which requires Hosted Zone
information?
    Type: String
  PrivateHostedZoneId:
    Description: The Route53 private zone ID to register the etcd targets with, such as
Z21IXYZABCZ2A4.
    Type: String
  PrivateHostedZoneName:
    Description: The Route53 zone to register the targets with, such as cluster.example.com. Omit
the trailing period.
    Type: String
  Master0Subnet:
    Description: The subnets, recommend private, to launch the master nodes into.
    Type: AWS::EC2::Subnet::Id
  Master1Subnet:
    Description: The subnets, recommend private, to launch the master nodes into.
    Type: AWS::EC2::Subnet::Id
  Master2Subnet:
    Description: The subnets, recommend private, to launch the master nodes into.
    Type: AWS::EC2::Subnet::Id
  MasterSecurityGroupId:
    Description: The master security group ID to associate with master nodes.
    Type: AWS::EC2::SecurityGroup::Id
  IgnitionLocation:
    Default: https://api-int.$CLUSTER_NAME.$DOMAIN:22623/config/master
    Description: Ignition config file location.
    Type: String
  CertificateAuthorities:
    Default: data:text/plain;charset=utf-8;base64,ABC...xYz==
    Description: Base64 encoded certificate authority string to use.
    Type: String
  MasterInstanceProfileName:
    Description: IAM profile to associate with master nodes.
    Type: String
```

```
MasterInstanceType:
  Default: m5.xlarge
  Type: String
  AllowedValues:
  - "m4.xlarge"
  - "m4.2xlarge"
  - "m4.4xlarge"
  - "m4.10xlarge"
  - "m4.16xlarge"
  - "m5.xlarge"
  - "m5.2xlarge"
  - "m5.4xlarge"
  - "m5.8xlarge"
  - "m5.12xlarge"
  - "m5.16xlarge"
  - "m5a.xlarge"
  - "m5a.2xlarge"
  - "m5a.4xlarge"
  - "m5a.8xlarge"
  - "m5a.10xlarge"
  - "m5a.16xlarge"
  - "c4.2xlarge"
  - "c4.4xlarge"
  - "c4.8xlarge"
  - "c5.2xlarge"
  - "c5.4xlarge"
  - "c5.9xlarge"
  - "c5.12xlarge"
  - "c5.18xlarge"
  - "c5.24xlarge"
  - "c5a.2xlarge"
  - "c5a.4xlarge"
  - "c5a.8xlarge"
  - "c5a.12xlarge"
  - "c5a.16xlarge"
  - "c5a.24xlarge"
  - "r4.xlarge"
  - "r4.2xlarge"
  - "r4.4xlarge"
  - "r4.8xlarge"
  - "r4.16xlarge"
  - "r5.xlarge"
  - "r5.2xlarge"
  - "r5.4xlarge"
  - "r5.8xlarge"
  - "r5.12xlarge"
  - "r5.16xlarge"
  - "r5.24xlarge"
  - "r5a.xlarge"
  - "r5a.2xlarge"
  - "r5a.4xlarge"
  - "r5a.8xlarge"
  - "r5a.12xlarge"
  - "r5a.16xlarge"
  - "r5a.24xlarge"
```

```
  AutoRegisterELB:
    Default: "yes"
    AllowedValues:
    - "yes"
    - "no"
    Description: Do you want to invoke NLB registration, which requires a Lambda ARN parameter?
    Type: String
  RegisterNlbIpTargetsLambdaArn:
    Description: ARN for NLB IP target registration lambda. Supply the value from the cluster
infrastructure or select "no" for AutoRegisterELB.
    Type: String
  ExternalApiTargetGroupArn:
    Description: ARN for external API load balancer target group. Supply the value from the cluster
infrastructure or select "no" for AutoRegisterELB.
    Type: String
  InternalApiTargetGroupArn:
    Description: ARN for internal API load balancer target group. Supply the value from the cluster
infrastructure or select "no" for AutoRegisterELB.
    Type: String
  InternalServiceTargetGroupArn:
    Description: ARN for internal service load balancer target group. Supply the value from the
cluster infrastructure or select "no" for AutoRegisterELB.
    Type: String

Metadata:
  AWS::CloudFormation::Interface:
    ParameterGroups:
    - Label:
        default: "Cluster Information"
      Parameters:
      - InfrastructureName
    - Label:
        default: "Host Information"
      Parameters:
      - MasterInstanceType
      - RhcosAmi
      - IgnitionLocation
      - CertificateAuthorities
      - MasterSecurityGroupId
      - MasterInstanceProfileName
    - Label:
        default: "Network Configuration"
      Parameters:
      - VpcId
      - AllowedBootstrapSshCidr
      - Master0Subnet
      - Master1Subnet
      - Master2Subnet
    - Label:
        default: "DNS"
      Parameters:
      - AutoRegisterDNS
      - PrivateHostedZoneName
      - PrivateHostedZoneId
    - Label:
        default: "Load Balancer Automation"
```

```
      Parameters:
      - AutoRegisterELB
      - RegisterNlbIpTargetsLambdaArn
      - ExternalApiTargetGroupArn
      - InternalApiTargetGroupArn
      - InternalServiceTargetGroupArn
    ParameterLabels:
      InfrastructureName:
        default: "Infrastructure Name"
      VpcId:
        default: "VPC ID"
      Master0Subnet:
        default: "Master-0 Subnet"
      Master1Subnet:
        default: "Master-1 Subnet"
      Master2Subnet:
        default: "Master-2 Subnet"
      MasterInstanceType:
        default: "Master Instance Type"
      MasterInstanceProfileName:
        default: "Master Instance Profile Name"
      RhcosAmi:
        default: "Red Hat Enterprise Linux CoreOS AMI ID"
      BootstrapIgnitionLocation:
        default: "Master Ignition Source"
      CertificateAuthorities:
        default: "Ignition CA String"
      MasterSecurityGroupId:
        default: "Master Security Group ID"
      AutoRegisterDNS:
        default: "Use Provided DNS Automation"
      AutoRegisterELB:
        default: "Use Provided ELB Automation"
      PrivateHostedZoneName:
        default: "Private Hosted Zone Name"
      PrivateHostedZoneId:
        default: "Private Hosted Zone ID"

Conditions:
  DoRegistration: !Equals ["yes", !Ref AutoRegisterELB]
  DoDns: !Equals ["yes", !Ref AutoRegisterDNS]

Resources:
  Master0:
    Type: AWS::EC2::Instance
    Properties:
      ImageId: !Ref RhcosAmi
      BlockDeviceMappings:
      - DeviceName: /dev/xvda
        Ebs:
          VolumeSize: "120"
          VolumeType: "gp2"
      IamInstanceProfile: !Ref MasterInstanceProfileName
      InstanceType: !Ref MasterInstanceType
      NetworkInterfaces:
      - AssociatePublicIpAddress: "false"
```

```
      DeviceIndex: "0"
      GroupSet:
      - !Ref "MasterSecurityGroupId"
      SubnetId: !Ref "Master0Subnet"
    UserData:
      Fn::Base64: !Sub
      - '{"ignition":{"config":{"merge":[{"source":"${SOURCE}"}]},"security":{"tls":
{"certificateAuthorities":[{"source":"${CA_BUNDLE}"}]}},"version":"3.1.0"}}'
      - {
        SOURCE: !Ref IgnitionLocation,
        CA_BUNDLE: !Ref CertificateAuthorities,
      }
    Tags:
    - Key: !Join ["", ["kubernetes.io/cluster/", !Ref InfrastructureName]]
      Value: "shared"

RegisterMaster0:
  Condition: DoRegistration
  Type: Custom::NLBRegister
  Properties:
    ServiceToken: !Ref RegisterNlbIpTargetsLambdaArn
    TargetArn: !Ref ExternalApiTargetGroupArn
    TargetIp: !GetAtt Master0.PrivateIp

RegisterMaster0InternalApiTarget:
  Condition: DoRegistration
  Type: Custom::NLBRegister
  Properties:
    ServiceToken: !Ref RegisterNlbIpTargetsLambdaArn
    TargetArn: !Ref InternalApiTargetGroupArn
    TargetIp: !GetAtt Master0.PrivateIp

RegisterMaster0InternalServiceTarget:
  Condition: DoRegistration
  Type: Custom::NLBRegister
  Properties:
    ServiceToken: !Ref RegisterNlbIpTargetsLambdaArn
    TargetArn: !Ref InternalServiceTargetGroupArn
    TargetIp: !GetAtt Master0.PrivateIp

Master1:
  Type: AWS::EC2::Instance
  Properties:
    ImageId: !Ref RhcosAmi
    BlockDeviceMappings:
    - DeviceName: /dev/xvda
      Ebs:
        VolumeSize: "120"
        VolumeType: "gp2"
    IamInstanceProfile: !Ref MasterInstanceProfileName
    InstanceType: !Ref MasterInstanceType
    NetworkInterfaces:
    - AssociatePublicIpAddress: "false"
      DeviceIndex: "0"
      GroupSet:
      - !Ref "MasterSecurityGroupId"
```

```
      SubnetId: !Ref "Master1Subnet"
    UserData:
      Fn::Base64: !Sub
      - '{"ignition":{"config":{"merge":[{"source":"${SOURCE}"}]},"security":{"tls":
{"certificateAuthorities":[{"source":"${CA_BUNDLE}"}]}},"version":"3.1.0"}}'
      - {
        SOURCE: !Ref IgnitionLocation,
        CA_BUNDLE: !Ref CertificateAuthorities,
      }
    Tags:
    - Key: !Join ["", ["kubernetes.io/cluster/", !Ref InfrastructureName]]
      Value: "shared"

RegisterMaster1:
  Condition: DoRegistration
  Type: Custom::NLBRegister
  Properties:
    ServiceToken: !Ref RegisterNlbIpTargetsLambdaArn
    TargetArn: !Ref ExternalApiTargetGroupArn
    TargetIp: !GetAtt Master1.PrivateIp

RegisterMaster1InternalApiTarget:
  Condition: DoRegistration
  Type: Custom::NLBRegister
  Properties:
    ServiceToken: !Ref RegisterNlbIpTargetsLambdaArn
    TargetArn: !Ref InternalApiTargetGroupArn
    TargetIp: !GetAtt Master1.PrivateIp

RegisterMaster1InternalServiceTarget:
  Condition: DoRegistration
  Type: Custom::NLBRegister
  Properties:
    ServiceToken: !Ref RegisterNlbIpTargetsLambdaArn
    TargetArn: !Ref InternalServiceTargetGroupArn
    TargetIp: !GetAtt Master1.PrivateIp

Master2:
  Type: AWS::EC2::Instance
  Properties:
    ImageId: !Ref RhcosAmi
    BlockDeviceMappings:
    - DeviceName: /dev/xvda
      Ebs:
        VolumeSize: "120"
        VolumeType: "gp2"
    IamInstanceProfile: !Ref MasterInstanceProfileName
    InstanceType: !Ref MasterInstanceType
    NetworkInterfaces:
    - AssociatePublicIpAddress: "false"
      DeviceIndex: "0"
      GroupSet:
      - !Ref "MasterSecurityGroupId"
      SubnetId: !Ref "Master2Subnet"
    UserData:
      Fn::Base64: !Sub
```

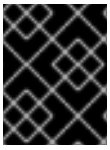
```
      - '{"ignition":{"config":{"merge":[{"source":"${SOURCE}"}]},"security":{"tls":
{"certificateAuthorities":[{"source":"${CA_BUNDLE}"}]}},"version":"3.1.0"}}'
      - {
        SOURCE: !Ref IgnitionLocation,
        CA_BUNDLE: !Ref CertificateAuthorities,
      }
    Tags:
    - Key: !Join ["", ["kubernetes.io/cluster/", !Ref InfrastructureName]]
      Value: "shared"

RegisterMaster2:
  Condition: DoRegistration
  Type: Custom::NLBRegister
  Properties:
    ServiceToken: !Ref RegisterNlbIpTargetsLambdaArn
    TargetArn: !Ref ExternalApiTargetGroupArn
    TargetIp: !GetAtt Master2.PrivateIp

RegisterMaster2InternalApiTarget:
  Condition: DoRegistration
  Type: Custom::NLBRegister
  Properties:
    ServiceToken: !Ref RegisterNlbIpTargetsLambdaArn
    TargetArn: !Ref InternalApiTargetGroupArn
    TargetIp: !GetAtt Master2.PrivateIp

RegisterMaster2InternalServiceTarget:
  Condition: DoRegistration
  Type: Custom::NLBRegister
  Properties:
    ServiceToken: !Ref RegisterNlbIpTargetsLambdaArn
    TargetArn: !Ref InternalServiceTargetGroupArn
    TargetIp: !GetAtt Master2.PrivateIp

EtcdSrvRecords:
  Condition: DoDns
  Type: AWS::Route53::RecordSet
  Properties:
    HostedZoneId: !Ref PrivateHostedZoneId
    Name: !Join [".", ["_etcd-server-ssl._tcp", !Ref PrivateHostedZoneName]]
    ResourceRecords:
    - !Join [
      " ",
      ["0 10 2380", !Join [".", ["etcd-0", !Ref PrivateHostedZoneName]]],
    ]
    - !Join [
      " ",
      ["0 10 2380", !Join [".", ["etcd-1", !Ref PrivateHostedZoneName]]],
    ]
    - !Join [
      " ",
      ["0 10 2380", !Join [".", ["etcd-2", !Ref PrivateHostedZoneName]]],
    ]
    TTL: 60
    Type: SRV
```

```
  Etcd0Record:
    Condition: DoDns
    Type: AWS::Route53::RecordSet
    Properties:
      HostedZoneId: !Ref PrivateHostedZoneId
      Name: !Join [".", ["etcd-0", !Ref PrivateHostedZoneName]]
      ResourceRecords:
      - !GetAtt Master0.PrivateIp
      TTL: 60
      Type: A

  Etcd1Record:
    Condition: DoDns
    Type: AWS::Route53::RecordSet
    Properties:
      HostedZoneId: !Ref PrivateHostedZoneId
      Name: !Join [".", ["etcd-1", !Ref PrivateHostedZoneName]]
      ResourceRecords:
      - !GetAtt Master1.PrivateIp
      TTL: 60
      Type: A

  Etcd2Record:
    Condition: DoDns
    Type: AWS::Route53::RecordSet
    Properties:
      HostedZoneId: !Ref PrivateHostedZoneId
      Name: !Join [".", ["etcd-2", !Ref PrivateHostedZoneName]]
      ResourceRecords:
      - !GetAtt Master2.PrivateIp
      TTL: 60
      Type: A

Outputs:
  PrivateIPs:
    Description: The control-plane node private IP addresses.
    Value:
      !Join [
        ",",
        [!GetAtt Master0.PrivateIp, !GetAtt Master1.PrivateIp, !GetAtt Master2.PrivateIp]
      ]
```

2.11.14. 在 AWS 中创建 worker 节点

您可以在 Amazon Web Services (AWS) 中创建 worker 节点，供集群使用。

您可以使用提供的 CloudFormation 模板和自定义参数文件创建代表 worker 节点的 AWS 资源堆栈。



重要

CloudFormation 模板会创建一个堆栈，它代表一个 worker 节点。您必须为每个 worker 节点创建一个堆栈。

注意

如果不使用提供的 CloudFormation 模板来创建 worker 节点，您必须检查提供的信息并手动创建基础架构。如果集群没有正确初始化，您可能需要联系红帽支持并提供您的安装日志。

先决条件

- 已配置了一个 AWS 帐户。
- 您可以通过运行 **aws configure**，将 AWS 密钥和区域添加到本地 AWS 配置集中。
- 已为集群生成 Ignition 配置文件。
- 您在 AWS 中创建并配置了 VPC 及相关子网。
- 您在 AWS 中创建并配置了 DNS、负载均衡器和监听程序。
- 您在 AWS 中创建了集群所需的安全组和角色。
- 已创建 bootstrap 机器。
- 已创建 control plane 机器。

流程

1. 创建一个 JSON 文件，其包含 CloudFormation 模板需要的参数值：

```
[
  {
    "ParameterKey": "InfrastructureName", 1
    "ParameterValue": "mycluster-<random_string>" 2
  },
  {
    "ParameterKey": "RhcosAmi", 3
    "ParameterValue": "ami-<random_string>" 4
  },
  {
    "ParameterKey": "Subnet", 5
    "ParameterValue": "subnet-<random_string>" 6
  },
  {
    "ParameterKey": "WorkerSecurityGroupId", 7
    "ParameterValue": "sg-<random_string>" 8
  },
  {
    "ParameterKey": "IgnitionLocation", 9
    "ParameterValue": "https://api-int.<cluster_name>.<domain_name>:22623/config/worker"
10
  },
  {
    "ParameterKey": "CertificateAuthorities", 11
    "ParameterValue": "" 12
  },
```

```
  {
    "ParameterKey": "WorkerInstanceProfileName", 13
    "ParameterValue": "" 14
  },
  {
    "ParameterKey": "WorkerInstanceType", 15
    "ParameterValue": "m4.large" 16
  }
]
```

1. 您的 Ignition 配置文件中为集群编码的集群基础架构名称。
2. 指定从 Ignition 配置文件元数据中提取的基础架构名称，其格式为 **<cluster-name>-<random-string>**。
3. 用于 worker 节点的当前 Red Hat Enterprise Linux CoreOS (RHCOS) AMI。
4. 指定 **AWS::EC2::Image::Id** 值。
5. 在其中启动 worker 节点的子网，最好是专用子网。
6. 从 DNS 和负载均衡的 CloudFormation 模板输出的 **PrivateSubnets** 值指定子网。
7. 与 worker 节点关联的 worker 安全组 ID。
8. 指定安全组和角色的 CloudFormation 模板输出的 **WorkerSecurityGroupId** 值。
9. 从中获取 bootstrap Ignition 配置文件的位置。
10. 指定生成的 Ignition 配置的位置，**https://api-int.<cluster_name>.<domain_name>:22623/config/worker**。
11. 要使用的 Base64 编码证书颁发机构字符串。
12. 指定安装目录下 **worker.ign** 文件中的值。这个值是一个长字符串，格式为 **data:text/plain;charset=utf-8;base64,ABC...xYz==**。
13. 与 worker 节点关联的 IAM 配置集。
14. 指定安全组和角色的 CloudFormation 模板输出的 **WorkerInstanceProfile** 参数值。
15. 用于 control plane 机器的 AWS 实例类型。
16. 允许的值：
    - **m4.large**
    - **m4.xlarge**
    - **m4.2xlarge**
    - **m4.4xlarge**
    - **m4.8xlarge**
    - **m4.10xlarge**

- **m4.16xlarge**
- **m5.large**
- **m5.xlarge**
- **m5.2xlarge**
- **m5.4xlarge**
- **m5.8xlarge**
- **m5.10xlarge**
- **m5.16xlarge**
- **m6i.xlarge**
- **c4.2xlarge**
- **c4.4xlarge**
- **c4.8xlarge**
- **r4.large**
- **r4.xlarge**
- **r4.2xlarge**
- **r4.4xlarge**
- **r4.8xlarge**
- **r4.16xlarge**

重要

如果您的区域中没有 **m4** 实例类型，例如 **eu-west-3**，请改为使用 **m5** 类型。

2. 复制 **worker 机器的 CloudFormation 模板**一节中的模板，并将它以 YAML 文件形式保存到计算机上。此模板描述了集群所需的网络对象和负载均衡器。
3. 如果您将 **m5** 实例类型指定为 **WorkerInstanceType** 的值，请将该实例类型添加到 CloudFormation 模板中的 **WorkerInstanceType.AllowedValues** 参数。
4. 启动 CloudFormation 模板，以创建代表 worker 节点的 AWS 资源堆栈：

重要

您必须在一行内输入命令。

```
$ aws cloudformation create-stack --stack-name <name> 1
    --template-body file://<template>.yaml \ 2
    --parameters file://<parameters>.json 3
```

1 **<name>** 是 CloudFormation 堆栈的名称，如 **cluster-worker-1**。如果您删除集群，则需要此堆栈的名称。

2 **<template>** 是您保存的 CloudFormation 模板 YAML 文件的相对路径和名称。

3 **<parameters>** 是 CloudFormation 参数 JSON 文件的相对路径和名称。

输出示例

```
arn:aws:cloudformation:us-east-1:269333783861:stack/cluster-worker-1/729ee301-1c2a-11eb-348f-sd9888c65b59
```

注意

CloudFormation 模板会创建一个堆栈，它代表一个 worker 节点。

5. 确认模板组件已存在：

```
$ aws cloudformation describe-stacks --stack-name <name>
```

6. 继续创建 worker 堆栈，直到为集群创建了充足的 worker 机器。您可以通过引用同一模板和参数文件并指定不同的堆栈名称来创建额外的 worker 堆栈。

重要

您必须至少创建两台 worker 机器，因此您必须创建至少两个使用此 CloudFormation 模板的堆栈。

2.11.14.1. worker 机器的 CloudFormation 模板

您可以使用以下 CloudFormation 模板来部署 OpenShift Container Platform 集群所需的 worker 机器。

例 2.48. worker 机器的 CloudFormation 模板

```
AWSTemplateFormatVersion: 2010-09-09
Description: Template for OpenShift Cluster Node Launch (EC2 worker instance)

Parameters:
  InfrastructureName:
    AllowedPattern: ^([a-zA-Z][a-zA-Z0-9\-]{0,26})$
    MaxLength: 27
    MinLength: 1
    ConstraintDescription: Infrastructure name must be alphanumeric, start with a letter, and have a
maximum of 27 characters.
    Description: A short, unique cluster ID used to tag nodes for the kubelet cloud provider.
    Type: String
  RhcosAmi:
    Description: Current Red Hat Enterprise Linux CoreOS AMI to use for bootstrap.
```

```
    Type: AWS::EC2::Image::Id
  Subnet:
    Description: The subnets, recommend private, to launch the master nodes into.
    Type: AWS::EC2::Subnet::Id
  WorkerSecurityGroupId:
    Description: The master security group ID to associate with master nodes.
    Type: AWS::EC2::SecurityGroup::Id
  IgnitionLocation:
    Default: https://api-int.$CLUSTER_NAME.$DOMAIN:22623/config/worker
    Description: Ignition config file location.
    Type: String
  CertificateAuthorities:
    Default: data:text/plain;charset=utf-8;base64,ABC...xYz==
    Description: Base64 encoded certificate authority string to use.
    Type: String
  WorkerInstanceProfileName:
    Description: IAM profile to associate with master nodes.
    Type: String
  WorkerInstanceType:
    Default: m5.large
    Type: String
    AllowedValues:
    - "m4.large"
    - "m4.xlarge"
    - "m4.2xlarge"
    - "m4.4xlarge"
    - "m4.10xlarge"
    - "m4.16xlarge"
    - "m5.large"
    - "m5.xlarge"
    - "m5.2xlarge"
    - "m5.4xlarge"
    - "m5.8xlarge"
    - "m5.12xlarge"
    - "m5.16xlarge"
    - "m5a.large"
    - "m5a.xlarge"
    - "m5a.2xlarge"
    - "m5a.4xlarge"
    - "m5a.8xlarge"
    - "m5a.10xlarge"
    - "m5a.16xlarge"
    - "c4.large"
    - "c4.xlarge"
    - "c4.2xlarge"
    - "c4.4xlarge"
    - "c4.8xlarge"
    - "c5.large"
    - "c5.xlarge"
    - "c5.2xlarge"
    - "c5.4xlarge"
    - "c5.9xlarge"
    - "c5.12xlarge"
    - "c5.18xlarge"
    - "c5.24xlarge"
    - "c5a.large"
```

```
      - "c5a.xlarge"
      - "c5a.2xlarge"
      - "c5a.4xlarge"
      - "c5a.8xlarge"
      - "c5a.12xlarge"
      - "c5a.16xlarge"
      - "c5a.24xlarge"
      - "r4.large"
      - "r4.xlarge"
      - "r4.2xlarge"
      - "r4.4xlarge"
      - "r4.8xlarge"
      - "r4.16xlarge"
      - "r5.large"
      - "r5.xlarge"
      - "r5.2xlarge"
      - "r5.4xlarge"
      - "r5.8xlarge"
      - "r5.12xlarge"
      - "r5.16xlarge"
      - "r5.24xlarge"
      - "r5a.large"
      - "r5a.xlarge"
      - "r5a.2xlarge"
      - "r5a.4xlarge"
      - "r5a.8xlarge"
      - "r5a.12xlarge"
      - "r5a.16xlarge"
      - "r5a.24xlarge"
      - "t3.large"
      - "t3.xlarge"
      - "t3.2xlarge"
      - "t3a.large"
      - "t3a.xlarge"
      - "t3a.2xlarge"

    Metadata:
      AWS::CloudFormation::Interface:
        ParameterGroups:
        - Label:
            default: "Cluster Information"
          Parameters:
          - InfrastructureName
        - Label:
            default: "Host Information"
          Parameters:
          - WorkerInstanceType
          - RhcosAmi
          - IgnitionLocation
          - CertificateAuthorities
          - WorkerSecurityGroupId
          - WorkerInstanceProfileName
        - Label:
            default: "Network Configuration"
          Parameters:
          - Subnet
```

```
    ParameterLabels:
      Subnet:
        default: "Subnet"
      InfrastructureName:
        default: "Infrastructure Name"
      WorkerInstanceType:
        default: "Worker Instance Type"
      WorkerInstanceProfileName:
        default: "Worker Instance Profile Name"
      RhcosAmi:
        default: "Red Hat Enterprise Linux CoreOS AMI ID"
      IgnitionLocation:
        default: "Worker Ignition Source"
      CertificateAuthorities:
        default: "Ignition CA String"
      WorkerSecurityGroupId:
        default: "Worker Security Group ID"

Resources:
  Worker0:
    Type: AWS::EC2::Instance
    Properties:
      ImageId: !Ref RhcosAmi
      BlockDeviceMappings:
      - DeviceName: /dev/xvda
        Ebs:
          VolumeSize: "120"
          VolumeType: "gp2"
      IamInstanceProfile: !Ref WorkerInstanceProfileName
      InstanceType: !Ref WorkerInstanceType
      NetworkInterfaces:
      - AssociatePublicIpAddress: "false"
        DeviceIndex: "0"
        GroupSet:
        - !Ref "WorkerSecurityGroupId"
        SubnetId: !Ref "Subnet"
      UserData:
        Fn::Base64: !Sub
        - '{"ignition":{"config":{"merge":[{"source":"${SOURCE}"}]},"security":{"tls":
{"certificateAuthorities":[{"source":"${CA_BUNDLE}"}]}},"version":"3.1.0"}}'
        - {
          SOURCE: !Ref IgnitionLocation,
          CA_BUNDLE: !Ref CertificateAuthorities,
        }
      Tags:
      - Key: !Join ["", ["kubernetes.io/cluster/", !Ref InfrastructureName]]
        Value: "shared"

Outputs:
  PrivateIP:
    Description: The compute node private IP address.
    Value: !GetAtt Worker0.PrivateIp
```

2.11.15. 使用用户置备的基础架构在 AWS 上初始化 bootstrap 序列

在 Amazon Web Services（AWS）中创建所有所需的基础架构后，您可以启动初始化 OpenShift Container Platform control plane 的 bootstrap 序列。

先决条件

- 已配置了一个 AWS 帐户。
- 您可以通过运行 **aws configure**，将 AWS 密钥和区域添加到本地 AWS 配置集中。
- 已为集群生成 Ignition 配置文件。
- 您在 AWS 中创建并配置了 VPC 及相关子网。
- 您在 AWS 中创建并配置了 DNS、负载均衡器和监听程序。
- 您在 AWS 中创建了集群所需的安全组和角色。
- 已创建 bootstrap 机器。
- 已创建 control plane 机器。
- 已创建 worker 节点。

流程

1. 更改为包含安装程序的目录，并启动初始化 OpenShift Container Platform control plane 的 bootstrap 过程：

```
$ ./openshift-install wait-for bootstrap-complete --dir <installation_directory> \ 1
    --log-level=info 2
```

1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

2 要查看不同的安装详情，请指定 **warn**、**debug** 或 **error**，而不要指定 **info**。

输出示例

```
INFO Waiting up to 20m0s for the Kubernetes API at
https://api.mycluster.example.com:6443...
INFO API v1.19.0+9f84db3 up
INFO Waiting up to 30m0s for bootstrapping to complete...
INFO It is now safe to remove the bootstrap resources
INFO Time elapsed: 1s
```

如果命令退出时没有 **FATAL** 警告，则 OpenShift Container Platform control plane 已被初始化。

注意

在 control plane 初始化后，它会设置计算节点，并以 Operator 的形式安装其他服务。

其他资源

- 如需了解在 OpenShift Container Platform 安装过程中监控安装、bootstrap 和 control plane 日志的详细信息，请参阅监控安装进度。
- 如需有关对 bootstrap 过程进行故障排除的信息，请参阅收集 bootstrap 节点诊断数据。

2.11.16. 使用 CLI 登录到集群

您可以通过导出集群 **kubeconfig** 文件，以默认系统用户身份登录集群。**kubeconfig** 文件包含关于集群的信息，供 CLI 用于将客户端连接到正确集群和 API 服务器。该文件特只适用于一个特定的集群，在 OpenShift Container Platform 安装过程中创建。

先决条件

- 已部署了 OpenShift Container Platform 集群。
- 已安装 **oc** CLI。

流程

1. 导出 **kubeadmin** 凭证：

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

2. 使用导出的配置，验证能否成功运行 **oc** 命令：

```
$ oc whoami
```

输出示例

```
system:admin
```

2.11.17. 批准机器的证书签名请求

将机器添加到集群时，会为您添加的每台机器生成两个待处理证书签名请求（CSR）。您必须确认这些 CSR 已获得批准，或根据需要自行批准。客户端请求必须首先被批准，然后是服务器请求。

先决条件

- 您已将机器添加到集群中。

流程

1. 确认集群可以识别这些机器：

```
$ oc get nodes
```

输出示例

```
NAME      STATUS    ROLES   AGE  VERSION
```

```
master-0  Ready    master  63m  v1.19.0
master-1  Ready    master  63m  v1.19.0
master-2  Ready    master  64m  v1.19.0
```

输出将列出您创建的所有机器。

注意

在一些 CSR 被批准前，以上输出可能不包括计算节点（也称为 worker 节点）。

2. 检查待处理的 CSR，并确保可以看到添加到集群中的每台机器都有 **Pending** 或 **Approved** 状态的客户端请求：

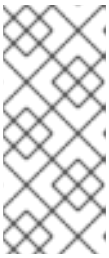
```
$ oc get csr
```

输出示例

```
NAME        AGE    REQUESTOR                                                   CONDITION
csr-8b2br   15m    system:serviceaccount:openshift-machine-config-operator:node-
bootstrapper   Pending
csr-8vnps   15m    system:serviceaccount:openshift-machine-config-operator:node-
bootstrapper   Pending
...
```

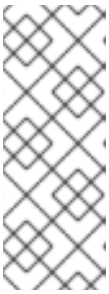
在本例中，两台机器加入了集群。您可能会在列表中看到更多已批准的 CSR。

3. 如果 CSR 没有获得批准，请在所添加机器的所有待处理 CSR 都处于 **Pending** 状态后，为您的集群机器批准这些 CSR：



注意

由于 CSR 会自动轮转，因此请在将机器添加到集群后一小时内批准您的 CSR。如果没有在一小时内批准，证书将会轮转，每个节点将会存在多个证书。您必须批准所有这些证书。批准客户端 CSR 后， Kubelet 为服务证书创建辅助 CSR，这需要手动批准。然后，如果 Kubelet 请求具有相同参数的新证书，则 **machine-approver** 会自动批准后续服务证书续订请求。



注意

对于在未启用机器 API 的平台中运行的集群，如裸机和其他用户置备的基础架构，必须采用一种方法自动批准 kubelet 提供证书请求（CSR）。如果没有批准请求，则 **oc exec**、**oc rsh** 和 **oc logs** 命令将无法成功，因为 API 服务器连接到 kubelet 时需要服务证书。与 Kubelet 端点联系的任何操作都需要此证书批准。这个方法必须监视新的 CSR，确认 CSR 由 **system:node** 或 **system:admin** 组中的 **node-bootstrapper** 服务帐户提交，并确认节点的身份。

- 若要单独批准，请对每个有效的 CSR 运行以下命令：

```
$ oc adm certificate approve <csr_name> 1
```

1 **<csr_name>** 是当前 CSR 列表中 CSR 的名称。

- 要批准所有待处理的 CSR，请运行以下命令：

```
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}{{"\n"}}
{{end}}{{end}}' | xargs --no-run-if-empty oc adm certificate approve
```

注意

在有些 CSR 被批准前，一些 Operator 可能无法使用。

4. 现在，您的客户端请求已被批准，您必须查看添加到集群中的每台机器的服务器请求：

```
$ oc get csr
```

输出示例

```
NAME        AGE     REQUESTOR                                                   CONDITION
csr-bfd72   5m26s   system:node:ip-10-0-50-126.us-east-2.compute.internal
Pending
csr-c57lv   5m26s   system:node:ip-10-0-95-157.us-east-2.compute.internal
Pending
...
```

5. 如果剩余的 CSR 没有被批准，且处于 **Pending** 状态，请批准集群机器的 CSR：

- 若要单独批准，请对每个有效的 CSR 运行以下命令：

```
$ oc adm certificate approve <csr_name> 1
```

1 **<csr_name>** 是当前 CSR 列表中 CSR 的名称。

- 要批准所有待处理的 CSR，请运行以下命令：

```
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}{{"\n"}}
{{end}}{{end}}' | xargs oc adm certificate approve
```

6. 批准所有客户端和服务器 CSR 后，器将处于 **Ready** 状态。运行以下命令验证：

```
$ oc get nodes
```

输出示例

```
NAME      STATUS    ROLES   AGE  VERSION
master-0  Ready     master  73m  v1.20.0
master-1  Ready     master  73m  v1.20.0
master-2  Ready     master  74m  v1.20.0
worker-0  Ready     worker  11m  v1.20.0
worker-1  Ready     worker  11m  v1.20.0
```

注意

批准服务器 CSR 后可能需要几分钟时间让机器转换为 **Ready** 状态。

其他信息

- 如需有关 CSR 的更多信息，请参阅证书签名请求。

2.11.18. 初始 Operator 配置

在 control plane 初始化后，您必须立即配置一些 Operator 以便它们都可用。

先决条件

- 您的 control plane 已初始化。

流程

1. 观察集群组件上线：

```
$ watch -n5 oc get clusteroperators
```

输出示例

```
NAME                                       VERSION AVAILABLE   PROGRESSING   DEGRADED
SINCE
authentication                             4.6.0   True        False         False      3h56m
cloud-credential                           4.6.0   True        False         False      29h
cluster-autoscaler                         4.6.0   True        False         False      29h
config-operator                            4.6.0   True        False         False      6h39m
console                                    4.6.0   True        False         False      3h59m
csi-snapshot-controller                    4.6.0   True        False         False      4h12m
dns                                        4.6.0   True        False         False      4h15m
etcd                                       4.6.0   True        False         False      29h
image-registry                             4.6.0   True        False         False      3h59m
ingress                                    4.6.0   True        False         False      4h30m
insights                                   4.6.0   True        False         False      29h
kube-apiserver                             4.6.0   True        False         False      29h
kube-controller-manager                    4.6.0   True        False         False      29h
kube-scheduler                             4.6.0   True        False         False      29h
kube-storage-version-migrator              4.6.0   True        False         False      4h2m
machine-api                                4.6.0   True        False         False      29h
machine-approver                           4.6.0   True        False         False      6h34m
machine-config                             4.6.0   True        False         False      3h56m
marketplace                                4.6.0   True        False         False      4h2m
monitoring                                 4.6.0   True        False         False      6h31m
network                                    4.6.0   True        False         False      29h
node-tuning                                4.6.0   True        False         False      4h30m
openshift-apiserver                        4.6.0   True        False         False      3h56m
openshift-controller-manager               4.6.0   True        False         False      4h36m
openshift-samples                          4.6.0   True        False         False      4h30m
operator-lifecycle-manager                 4.6.0   True        False         False      29h
operator-lifecycle-manager-catalog         4.6.0   True        False         False      29h
operator-lifecycle-manager-packageserver   4.6.0   True        False         False      3h59m
service-ca                                 4.6.0   True        False         False      29h
storage                                    4.6.0   True        False         False      4h30m
```

2. 配置不可用的 Operator。

2.11.18.1. 禁用默认的 OperatorHub 源

在 OpenShift Container Platform 安装过程中，默认为 OperatorHub 配置由红帽和社区项目提供的源内容的 operator 目录。在受限网络环境中，必须以集群管理员身份禁用默认目录。

流程

- 通过在 **OperatorHub** 对象中添加 **disableAllDefaultSources: true** 来禁用默认目录的源：

```
$ oc patch OperatorHub cluster --type json \
    -p '[{"op": "add", "path": "/spec/disableAllDefaultSources", "value": true}]'
```

提示

或者，您可以使用 Web 控制台管理目录源。在 **Administration → Cluster Settings → Global Configuration → OperatorHub** 页面中，点 **Sources** 选项卡，其中可创建、删除、禁用和启用单独的源。

2.11.18.2. 镜像 registry 存储配置

Amazon Web Services 提供默认存储，这意味着 Image Registry Operator 在安装后可用。但是，如果 Registry Operator 无法创建 S3 存储桶并自动配置存储，您需要手工配置 registry 存储。

示配置生产集群所需的持久性卷的说明。如果适用，显示有关将空目录配置为存储位置的说明，该位置只可用于非生产集群。

另外还提供了在升级过程中使用 **Recreate** rollout 策略来允许镜像 registry 使用块存储类型的说明。

2.11.18.2.1. 为使用用户置备的基础架构的 AWS 配置 registry 存储

在安装过程中，使用您的云凭据就可以创建一个 Amazon S3 存储桶，Registry Operator 将会自动配置存储。

如果 Registry Operator 无法创建 S3 存储桶或自动配置存储，您可以按照以下流程创建 S3 存储桶并配置存储。

先决条件

- 在带有用户置备的基础架构的 AWS 上有一个集群。
- 对于 Amazon S3 存储，secret 应该包含以下两个键：
  - **REGISTRY_STORAGE_S3_ACCESSKEY**
  - **REGISTRY_STORAGE_S3_SECRETKEY**

流程

如果 Registry Operator 无法创建 S3 存储桶并自动配置存储，请进行以下操作。

1. 设置一个 Bucket Lifecycle Policy 用来终止已有一天之久的未完成的分段上传操作。
2. 在**configs.imageregistry.operator.openshift.io/cluster**中中输入存储配置：

```
$ oc edit configs.imageregistry.operator.openshift.io/cluster
```

配置示例

```
storage:
  s3:
    bucket: <bucket-name>
    region: <region-name>
```

警告

为了保护 AWS 中 registry 镜像的安全，阻止对 S3 存储桶的公共访问。

2.11.18.2.2. 在非生产集群中配置镜像 registry 存储

您必须为 Image Registry Operator 配置存储。对于非生产集群，您可以将镜像 registry 设置为空目录。如果您这样做，重启 registry 后会丢失所有镜像。

流程

- 将镜像 registry 存储设置为空目录：

```
$ oc patch configs.imageregistry.operator.openshift.io cluster --type merge --patch '{"spec":{"storage":{"emptyDir":{}}}}'
```

警告

仅可为非生产集群配置这个选项。

如果在 Image Registry Operator 初始化其组件前运行此命令，**oc patch** 命令会失败并显示以下错误：

```
Error from server (NotFound): configs.imageregistry.operator.openshift.io "cluster" not found
```

等待几分钟，然后再次运行该命令。

2.11.19. 删除 bootstrap 资源：

完成集群的初始 Operator 配置后，从 Amazon Web Services (AWS) 中删除 bootstrap 资源。

先决条件

- 已为集群完成初始的 Operator 配置。

流程

1. 删除 bootstrap 资源。如果您使用了 CloudFormation 模板，请删除其堆栈：

   - 使用 AWS CLI 删除堆栈：

     ```
     $ aws cloudformation delete-stack --stack-name <name> 1
     ```

     1 **<name>** 是 bootstrap 堆栈的名称。

   - 使用 AWS CloudFormation 控制台删除堆栈。

2.11.20. 创建 Ingress DNS 记录

如果您删除了 DNS 区配置，请手动创建指向 Ingress 负载均衡器的 DNS 记录。您可以创建一个 wildcard 记录或具体的记录。以下流程使用了 A 记录，但您可以使用其他所需记录类型，如 CNAME 或别名。

先决条件

- 已在 Amazon Web Services (AWS) 上安装了使用您置备的基础架构的 OpenShift Container Platform 集群。
- 已安装 OpenShift CLI（**oc**）。
- 安装了 **jq** 软件包。
- 您下载了 AWS CLI 并安装到您的计算机上。请参阅使用捆绑安装程序（Linux、macOS 或 Unix）安装 AWS CLI的文档。

流程

1. 决定要创建的路由。

   - 要创建一个 wildcard 记录，请使用 ***.apps.<cluster_name>.<domain_name>**，其中 **<cluster_name>** 是集群名称，**<domain_name>** 是 OpenShift Container Platform 集群的 Route 53 基域。
   - 要创建特定的记录，您必须为集群使用的每个路由创建一个记录，如下所示：

     ```
     $ oc get --all-namespaces -o jsonpath='{range .items[*]}{range .status.ingress[*]}{.host}
     {"\n"}{end}{end}' routes
     ```

     输出示例

     ```
     oauth-openshift.apps.<cluster_name>.<domain_name>
     console-openshift-console.apps.<cluster_name>.<domain_name>
     downloads-openshift-console.apps.<cluster_name>.<domain_name>
     alertmanager-main-openshift-monitoring.apps.<cluster_name>.<domain_name>
     grafana-openshift-monitoring.apps.<cluster_name>.<domain_name>
     prometheus-k8s-openshift-monitoring.apps.<cluster_name>.<domain_name>
     ```

2. 获取 Ingress Operator 负载均衡器状态，并记录其使用的外部 IP 地址值，如 **EXTERNAL-IP** 列所示：

   ```
   $ oc -n openshift-ingress get service router-default
   ```

输出示例

```
NAME              TYPE           CLUSTER-IP     EXTERNAL-IP                          PORT(S)
AGE
router-default    LoadBalancer   172.30.62.215  ab3...28.us-east-2.elb.amazonaws.com
80:31499/TCP,443:30693/TCP   5m
```

3. 为负载均衡器定位托管区 ID：

```
$ aws elb describe-load-balancers | jq -r '.LoadBalancerDescriptions[] | select(.DNSName ==
"<external_ip>").CanonicalHostedZoneNameID' 1
```

1 对于 **<external_ip>**，请指定您获取的 Ingress Operator 负载均衡器的外部 IP 地址值。

输出示例

```
Z3AADJGX6KTTL2
```

这个命令的输出是负载均衡器托管区 ID。

4. 获取集群域的公共托管区 ID：

```
$ aws route53 list-hosted-zones-by-name \
        --dns-name "<domain_name>" \ 1
        --query 'HostedZones[? Config.PrivateZone != `true` && Name ==
`<domain_name>.`].Id' 2
        --output text
```

1 2 对于 **<domain_name>**，请为 OpenShift Container Platform 集群指定 Route 53 基域。

输出示例

```
/hostedzone/Z3URY6TWQ91KVV
```

命令输出中会显示您的域的公共托管区 ID。在本例中是 **Z3URY6TWQ91KVV**。

5. 在您的私有区中添加别名记录：

```
$ aws route53 change-resource-record-sets --hosted-zone-id "<private_hosted_zone_id>" --
change-batch '{ 1
>   "Changes": [
>     {
>       "Action": "CREATE",
>       "ResourceRecordSet": {
>         "Name": "\\052.apps.<cluster_domain>", 2
>         "Type": "A",
>         "AliasTarget":{
>           "HostedZoneId": "<hosted_zone_id>", 3
>           "DNSName": "<external_ip>.", 4
>           "EvaluateTargetHealth": false
>         }
```

```
>     }
>   }
> ]
>}'
```

1. 对于 **<private_hosted_zone_id>**，指定 DNS 和负载均衡的 CloudFormation 模板输出的值。
2. 对于 **<cluster_domain>**，请指定用于 OpenShift Container Platform 集群的域或子域。
3. 对于 **<hosted_zone_id>**，请为您获得的负载均衡器指定公共托管区 ID。
4. 对于 **<external_ip>**，请指定 Ingress Operator 负载均衡器的外部 IP 地址值。请确定在该参数值中包含最后的句点（.）。

6. 在您的公共区中添加记录：

```
$ aws route53 change-resource-record-sets --hosted-zone-id "<public_hosted_zone_id>"" --
change-batch '{ 1
>   "Changes": [
>     {
>       "Action": "CREATE",
>       "ResourceRecordSet": {
>         "Name": "\\052.apps.<cluster_domain>", 2
>         "Type": "A",
>         "AliasTarget":{
>           "HostedZoneId": "<hosted_zone_id>", 3
>           "DNSName": "<external_ip>.", 4
>           "EvaluateTargetHealth": false
>         }
>       }
>     }
>   ]
>}'
```

1. 对于 **<public_hosted_zone_id>**，请为您的域指定公共托管区。
2. 对于 **<cluster_domain>**，请指定用于 OpenShift Container Platform 集群的域或子域。
3. 对于 **<hosted_zone_id>**，请为您获得的负载均衡器指定公共托管区 ID。
4. 对于 **<external_ip>**，请指定 Ingress Operator 负载均衡器的外部 IP 地址值。请确定在该参数值中包含最后的句点（.）。

2.11.21. 在用户置备的基础架构上完成 AWS 安装

在用户置备的基础架构 Amazon Web Service (AWS) 上启动 OpenShift Container Platform 安装后，监视进程并等待安装完成。

先决条件

- 您在用户置备的 AWS 基础架构上为 OpenShift Container Platform 集群删除了 bootstrap 节点。

- 已安装 **oc** CLI。

流程

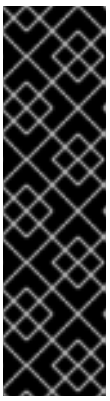
1. 在包含安装程序的目录中完成集群安装：

```
$ ./openshift-install --dir <installation_directory> wait-for install-complete 1
```

1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

输出示例

```
INFO Waiting up to 40m0s for the cluster at https://api.mycluster.example.com:6443 to
initialize...
INFO Waiting up to 10m0s for the openshift-console route to be created...
INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run 'export
KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'
INFO Access the OpenShift web-console here: https://console-openshift-
console.apps.mycluster.example.com
INFO Login to the console with user: "kubeadmin", and password: "4vYBz-Fe5en-ymBEc-
Wt6NL"
INFO Time elapsed: 1s
```



重要

- 安装程序生成的 Ignition 配置文件包含在 24 小时后过期的证书，然后在过期时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外情况是，您需要手动批准待处理的 **node-bootstrapper** 证书签名请求（CSR）来恢复 kubelet 证书。如需更多信息，请参阅*从过期的 control plane 证书中恢复*的文档。
- 建议您在生成 12 小时后使用 Ignition 配置文件，因为集群安装后 24 小时证书从 16 小时轮转至 22 小时。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中运行证书更新时避免安装失败。

2. 在 Cluster registration 页面注册您的集群。

2.11.22. 使用 Web 控制台登录到集群

kubeadmin 用户默认在 OpenShift Container Platform 安装后存在。您可以使用 OpenShift Container Platform Web 控制台以 **kubeadmin** 用户身份登录集群。

先决条件

- 有访问安装主机的访问权限。
- 您完成了集群安装，所有集群 Operator 都可用。

流程

1. 从安装主机上的 **kubeadmin -password** 文件中获取 kubeadmin 用户的密码：

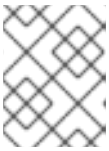
```
$ cat <installation_directory>/auth/kubeadmin-password
```

注意

另外，您还可以从安装主机上的 **<installation_directory>/.openshift_install.log** 日志文件获取 **kubeadmin** 密码。

2. 列出 OpenShift Container Platform Web 控制台路由：

```
$ oc get routes -n openshift-console | grep 'console-openshift'
```



注意

另外，您还可以从安装主机上的 **<installation_directory>/.openshift_install.log 日志** 文件获取 OpenShift Container Platform 路由。

输出示例

```
console     console-openshift-console.apps.<cluster_name>.<base_domain>          console
https   reencrypt/Redirect   None
```

3. 在 Web 浏览器中导航到上一命令输出中包括的路由，以 **kubeadmin** 用户身份登录。

其他资源

- 如需有关访问和了解 OpenShift Container Platform Web 控制台的更多信息，请参阅访问 Web 控制台。

2.11.23. OpenShift Container Platform 的 Telemetry 访问

在 OpenShift Container Platform 4.6 中，默认运行的 Telemetry 服务提供有关集群健康状况和成功更新的指标，需要访问互联网。如果您的集群连接到互联网，Telemetry 会自动运行，而且集群会注册到 OpenShift Cluster Manager。

确认 OpenShift Cluster Manager 清单正确后，可以由 Telemetry 自动维护，也可以使用 OpenShift Cluster Manager 手动维护，使用订阅监控来跟踪帐户或多集群级别的 OpenShift Container Platform 订阅。

其他资源

- 有关 Telemetry 服务的更多信息，请参阅关于远程健康监控。

2.11.24. 其他资源

- 如需有关 AWS CloudFormation 堆栈的更多信息，请参阅 AWS 文档中的使用堆栈。

2.11.25. 后续步骤

- 验证安装。
- 自定义集群。

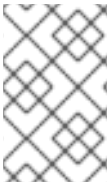
- 为 Cluster Samples Operator 和 **must-gather** 工具配置镜像流。
- 了解如何在受限网络中使用 Operator Lifecycle Manager（OLM）。
- 如果您用来安装集群的镜像 registry 具有一个可信任的 CA，通过配置额外的信任存储将其添加到集群中。
- 如果需要，您可以选择不使用远程健康报告。
- 如果需要，您可以删除云供应商凭证。

2.12. 在 AWS 上卸载集群

您可以删除部署到 Amazon Web Services (AWS) 的集群。

2.12.1. 删除使用安装程序置备的基础架构的集群

您可以从云中删除使用安装程序置备的基础架构的集群。



注意

卸载后，检查云供应商是否有没有被正确移除的资源，特别是 User Provisioned Infrastructure（UPI）集群。可能存在安装程序没有创建的资源，或者安装程序无法访问的资源。

先决条件

- 有部署集群时所用的安装程序副本。
- 有创建集群时安装程序所生成的文件。

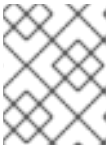
流程

1. 在用来安装集群的计算机中包含安装程序的目录中，运行以下命令：

```
$ ./openshift-install destroy cluster \
--dir <installation_directory> --log-level info 1 2
```

1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

2 要查看不同的详情，请指定 **warn**、**debug** 或 **error**，而不要指定 **info**。



注意

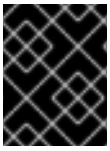
您必须为集群指定包含集群定义文件的目录。安装程序需要此目录中的 **metadata.json** 文件来删除集群。

2. 可选：删除 **<installation_directory>** 目录和 OpenShift Container Platform 安装程序。

第 3 章 在 AZURE 上安装

3.1. 配置 AZURE 帐户

在安装 OpenShift Container Platform 之前，您必须配置 Microsoft Azure 帐户。



重要

所有通过公共端点提供的 Azure 资源均存在资源名称的限制，您无法创建使用某些名称的资源。如需 Azure 限制词语列表，请参阅 Azure 文档中的解决保留资源名称错误。

3.1.1. Azure 帐户限值

OpenShift Container Platform 集群使用诸多 Microsoft Azure 组件，默认的 Azure 订阅和服务限值、配额和约束会影响您安装 OpenShift Container Platform 集群的能力。

重要

默认的限制因服务类别的不同（如 Free Trial 或 Pay-As-You-Go）以及系列的不同（如 Dv2、F 或 G）而有所不同。例如，对于 Enterprise Agreement 订阅的默认限制是 350 个内核。

在 Azure 上安装默认集群前，请检查您的订阅类型的限制，如有必要，请提高帐户的配额限制。

下表总结了 Azure 组件，它们的限值会影响您安装和运行 OpenShift Container Platform 集群的能力。

组件	默认所需的组件数	默认 Azure 限值	描述

组件	默认所需的组件数	默认 Azure 限值	描述
vCPU	40	每个区域 20 个	默认集群需要 40 个 vCPU，因此您必须提高帐户限值。 默认情况下，每个集群创建以下实例： ● 一台 Bootstrap 机器，在安装后删除 ● 三个 control plane 机器 ● 三个计算（compute）机器 由于 Bootstrap 机器使用 **Standard_D4s_v3** 机器（使用 4 个 vCPU），control plane 机器使用 **Standard_D8s_v3** 虚拟机（8 个 vCPU），并且 worker 机器使用 **Standard_D4s_v3** 虚拟机（4 个 vCPU），因此默认集群需要 40 个 vCPU。bootstrap 节点 VM（使用 4 个 vCPU）只在安装过程中使用。 若要部署更多 worker 节点、启用自动扩展、部署大型工作负载或使用不同的实例类型，您必须进一步提高帐户的 vCPU 限值，以确保集群可以部署您需要的机器。 默认情况下，安装程序将 control plane 和 compute 机器分布到一个区域中的所有可用区。要确保集群的高可用性，请选择至少含有三个可用区的区域。如果您的区域包含的可用区少于三个，安装程序将在可用区中放置多台 control plane 机器。
OS Disk	7		虚拟机 OS 磁盘必须能够保持最低 5000 IOPS/200MBps 的吞吐量。此吞吐量可以通过至少 1 TiB Premium SSD (P30) 提供。在 Azure 中，磁盘性能直接依赖于 SSD 磁盘大小，因此要达到 **Standard_D8s_v3** 支持的吞吐量，或其他类似的机器类型，目标为 5000 IOPS，则至少需要一个 P30 磁盘。 主机缓存必须设置为 **ReadOnly** 以获得低读取延迟和高读取 IOPS 和吞吐量。从缓存执行的读取可能存在于虚拟机内存或本地 SSD 磁盘中，比数据磁盘的读取速度要快得多，因为数据磁盘位于 blob 存储中。
VNet	1	每个区域 1000 个	每个默认集群都需要一个虚拟网络 (VNet)，此网络包括两个子网。
网络接口	6	每个区域 65,536 个	每个默认集群都需要六个网络接口。如果您要创建更多机器或者您部署的工作负载要创建负载均衡器，则集群会使用更多的网络接口。

组件	默认所需的组件数	默认 Azure 限值	描述
网络安全组	2	5000	每个默认集群为 VNet 中的每个子网创建网络安全组。默认集群为 control plane 和计算节点子网创建网络安全组： **control plane**：允许从任何位置通过端口 6443 访问 control plane 机器 **node**：允许从互联网通过端口 80 和 443 访问 worker 节点
网络负载均衡器	3	每个区域 1000 个	每个集群都会创建以下负载均衡器： **default**：用于在 worker 机器之间对端口 80 和 443 的请求进行负载均衡的公共 IP 地址 **internal**：用于在 control plane 机器之间对端口 6443 和 22623 的请求进行负载均衡的专用 IP 地址 **external**：用于在 control plane 机器之间对端口 6443 的请求进行负载均衡的公共 IP 地址 如果您的应用程序创建了更多的 Kubernetes **LoadBalancer** 服务对象，您的集群会使用更多的负载均衡器。
公共 IP 地址	3		两个公共负载均衡器各自使用一个公共 IP 地址。bootstrap 机器也使用一个公共 IP 地址，以便您可以在安装期间通过 SSH 连接到该机器来进行故障排除。bootstrap 节点的 IP 地址仅在安装过程中使用。
专用 IP 地址	7		内部负载均衡器、三台 control plane 机器中的每一台以及三台 worker 机器中的每一台各自使用一个专用 IP 地址。
Spot VM vCPU（可选）	0 如果配置 spot 虚拟机，您的集群必须为每个计算节点有两个 spot VM vCPU。	每个区域 20 个	这是可选组件。要使用 spot 虚拟机，您必须将 Azure 默认限值增加到集群中至少有两倍的计算节点数量。  **注意** 不建议将 spot 虚拟机用于 control plane 节点。

3.1.2. 在 Azure 中配置公共 DNS 区

要安装 OpenShift Container Platform，您使用的 Microsoft Azure 帐户必须在帐户中具有一个专用的公共托管 DNS 区。此区域必须对域具有权威。此服务为集群外部连接提供集群 DNS 解析和名称查询。

流程

1. 标识您的域或子域，以及注册商（registrar）。您可以转移现有的域和注册商，或通过 Azure 或其他来源获取新的域和注册商。

注意

如需通过 Azure 购买域的更多信息，请参阅 Azure 文档中的购买 Azure 应用服务的自定义域名。

2. 如果您使用现有的域和注册商，请将其 DNS 迁移到 Azure。请参阅 Azure 文档中的将活动 DNS 名称迁移到 Azure 应用服务。

3. 为您的域配置 DNS。按照 Azure 文档中教程：在 Azure DNS 中托管域部分里的步骤，为您的域或子域创建一个公共托管区，提取新的权威名称服务器，并更新您的域使用的名称服务器的注册商记录。
使用合适的根域（如 **openshiftcorp.com**）或子域（如 **clusters.openshiftcorp.com**）。

4. 如果您使用子域，请按照您公司的流程将其委派记录添加到父域。

3.1.3. 提高 Azure 帐户限值

要提高帐户限值，请在 Azure 门户上提交支持请求。

注意

每一支持请求只能提高一种类型的配额。

流程

1. 从 Azure 门户，点击左下角的 **Help + suport**。

2. 点击 **New support request**，然后选择所需的值：

   a. 从 **Issue type** 列表中，选择 **Service and subscription limits (quotas)**。

   b. 从 **Subscription** 列表中，选择要修改的订阅。

   c. 从 **Quota type** 列表中，选择要提高的配额。例如，选择 **Compute-VM (cores-vCPUs) subscription limit increases** 以增加 vCPU 的数量，这是安装集群所必须的。

   d. 点击 **Next: Solutions**。

3. 在 **Problem Details** 页面中，提供您要提高配额所需的信息：

   a. 点击 **Provide details**，然后在 **Quota details** 窗口中提供所需的详情。

   b. 在 SUPPORT METHOD 和 CONTACT INFO 部分中，提供问题严重性和您的联系详情。

4. 点击 **Next: Review + create**，然后点击 **Create**。

3.1.4. 所需的 Azure 角色

OpenShift Container Platform 需要一个服务主体，以便可以管理 Microsoft Azure 资源。在创建服务主体前，您的 Azure 帐户订阅必须具有以下角色：

- **User Access Administrator**
- **所有者**

要在 Azure 门户上设置角色，请参阅 Azure 文档中的使用 RBAC 和 Azure 门户管理对 Azure 资源的访问。

3.1.5. 创建服务主体

由于 OpenShift Container Platform 及其安装程序必须通过 Azure Resource Manager 创建 Microsoft Azure 资源，因此您必须创建一个能代表它的服务主体。

先决条件

- 安装或更新 Azure CLI。
- 安装**jq**软件包。
- 您的 Azure 帐户具有您所用订阅所需的角色。

流程

1. 登录 Azure CLI：

```
$ az login
```

   在 Web 控制台中，使用您的凭证登录 Azure。

2. 如果您的 Azure 帐户使用订阅，请确保使用正确的订阅。
   a. 查看可用帐户列表并记录您要用于集群的订阅的 **tenantId** 值：

```
$ az account list --refresh
```

   输出示例

```
[
  {
    "cloudName": "AzureCloud",
    "id": "9bab1460-96d5-40b3-a78e-17b15e978a80",
    "isDefault": true,
    "name": "Subscription Name",
    "state": "Enabled",
    "tenantId": "6057c7e9-b3ae-489d-a54e-de3f6bf6a8ee",
    "user": {
      "name": "you@example.com",
      "type": "user"
    }
  }
]
```

b. 查看您的活跃帐户详情，确认 **tenantId** 值与您要使用的订阅匹配：

```
$ az account show
```

输出示例

```
{
  "environmentName": "AzureCloud",
  "id": "9bab1460-96d5-40b3-a78e-17b15e978a80",
  "isDefault": true,
  "name": "Subscription Name",
  "state": "Enabled",
  "tenantId": "6057c7e9-b3ae-489d-a54e-de3f6bf6a8ee", (1)
  "user": {
    "name": "you@example.com",
    "type": "user"
  }
}
```

(1) 确定 **tenantId** 参数的值是正确订阅的 UUID。

c. 如果您使用的订阅不正确，请更改活跃的订阅：

```
$ az account set -s <id> (1)
```

(1) 替换您要用于 **<id>** 的订阅的 **id** 值。

d. 如果您更改了活跃订阅，请重新显示您的帐户信息：

```
$ az account show
```

输出示例

```
{
  "environmentName": "AzureCloud",
  "id": "33212d16-bdf6-45cb-b038-f6565b61edda",
  "isDefault": true,
  "name": "Subscription Name",
  "state": "Enabled",
  "tenantId": "8049c7e9-c3de-762d-a54e-dc3f6be6a7ee",
  "user": {
    "name": "you@example.com",
    "type": "user"
  }
}
```

3. 记录前面输出中 **tenantId** 和 **id** 参数的值。OpenShift Container Platform 安装过程中需要这些值。

4. 为您的帐户创建服务主体：

```
$ az ad sp create-for-rbac --role Contributor --name <service_principal> (1)
```

```
```

1 将 **<service_principal>** 替换要您要分配给服务主体的名称。

输出示例

```
Changing "<service_principal>" to a valid URI of "http://<service_principal>", which is the
required format used for service principal names
Retrying role assignment creation: 1/36
Retrying role assignment creation: 2/36
Retrying role assignment creation: 3/36
Retrying role assignment creation: 4/36
{
  "appId": "8bd0d04d-0ac2-43a8-928d-705c598c6956",
  "displayName": "<service_principal>",
  "name": "http://<service_principal>",
  "password": "ac461d78-bf4b-4387-ad16-7e32e328aec6",
  "tenant": "6048c7e9-b2ad-488d-a54e-dc3f6be6a7ee"
}
```

5. 记录前面输出中 **appId** 和 **password** 参数的值。OpenShift Container Platform 安装过程中需要这些值。

6. 为服务主体授予额外权限。

   - 您必须始终将 **Contributor** 和 **User Access Administrator** 角色添加到应用程序注册服务主体中，以便集群可以为组件分配凭证。
   - 要以 *mint 模式*操作 Cloud Credential Operator（CCO），应用程序注册服务主体还需要 **Azure Active Directory Graph/Application.ReadWrite.OwnedBy** API 权限。
   - 要以 *passthrough 模式* 操作 CCO，应用程序注册服务主体不需要额外的 API 权限。

   如需有关 CCO 模式的更多信息，请参阅 **Red Hat Operator** 参考内容中的 **Cloud Credential Operator** 条目。

   a. 要分配 **User Access Administrator** 角色，请运行以下命令：

   ```
   $ az role assignment create --role "User Access Administrator" \
       --assignee-object-id $(az ad sp list --filter "appId eq '<appId>'" \
         | jq '.[0].id' -r) 1
   ```

   1 将 **<appId>** 替换为服务器主体的 **appId** 参数值。

   b. 要分配 **Azure Active Directory Graph** 权限，请运行以下命令：

   ```
   $ az ad app permission add --id <appId> \ 1
       --api 00000002-0000-0000-c000-000000000000 \
       --api-permissions 824c81eb-e3f8-4ee6-8f6d-de7f50d565b7=Role
   ```

   1 将 **<appId>** 替换为服务器主体的 **appId** 参数值。

   输出示例

   ```
   -
   ```

> Invoking "az ad app permission grant --id 46d33abc-b8a3-46d8-8c84-f0fd58177435 --api 00000002-0000-0000-c000-000000000000" is needed to make the change effective

如需进一步了解可通过此命令授予的具体权限，请参阅 Windows Azure Active Directory 权限的 GUID 表。

c. 批准权限请求。如果您的帐户没有 Azure Active Directory 租户管理员角色，请按照您的组织的准则请租户管理员批准您的权限请求。

```
$ az ad app permission grant --id <appId> \ 1
    --api 00000002-0000-0000-c000-000000000000
```

1 将 **<appId>** 替换为服务器主体的 **appId** 参数值。

3.1.6. 支持的 Azure 区域

安装程序会根据您的订阅动态地生成可用的 Microsoft Azure 区域列表。OpenShift Container Platform 4.6.1 中已测试并验证了以下 Azure 区域：

支持的 Azure 公共区域

- **australiacentral** (Australia Central)
- **australiaeast** (Australia East)
- **australiasoutheast** (Australia South East)
- **brazilsouth** (Brazil South)
- **canadacentral** (Canada Central)
- **canadaeast** (Canada East)
- **centralindia** (Central India)
- **centralus** (Central US)
- **eastasia** (East Asia)
- **eastus** (East US)
- **eastus2** (East US 2)
- **francecentral** (France Central)
- **germanywestcentral** (Germany West Central)
- **japaneast** (Japan East)
- **japanwest** (Japan West)
- **koreacentral** (Korea Central)
- **koreasouth** (Korea South)

- **northcentralus** (North Central US)
- **northeurope** (North Europe)
- **norwayeast** (Norway East)
- **southafricanorth** (South Africa North)
- **southcentralus** (South Central US)
- **southeastasia** (Southeast Asia)
- **southindia** (South India)
- **switzerlandnorth** (Switzerland North)
- **uaenorth** (UAE North)
- **uksouth** (UK South)
- **ukwest** (UK West)
- **westcentralus** (West Central US)
- **westeurope** (West Europe)
- **westindia** (West India)
- **westus** (West US)
- **westus2** (West US 2)

支持的 Azure 政府区域
OpenShift Container Platform 4.6 添加了对以下 Microsoft Azure Government（MAG）区域的支持：

- **usgovtexas** (US Gov Texas)
- **usgovvirginia** (US Gov Virginia)

您可以参阅 Azure 文档来了解与所有可用 MAG 区域的信息。其他 MAG 区域应该可以与 OpenShift Container Platform 一起工作，但并没有经过测试。

3.1.7. 后续步骤

- 在 Azure 上安装 OpenShift Container Platform 集群。您可以安装自定义集群，或使用默认选项快速安装集群。

3.2. 为 AZURE 手动创建 IAM

在无法访问云身份和访问管理（IAM）API 的环境中，或者管理员更不希望将管理员级别的凭证 secret 存储在集群 **kube-system** 命名空间中时，可以在安装前将 Cloud Credential Operator（CCO）放入手动模式。

3.2.1. 在 kube-system 项目中存储管理员级别的 secret 的替代方案

Cloud Credential Operator（CCO） 将云供应商凭证作为 Kubernetes 自定义资源定义 (CRD) 进行管理。您可以通过在 **install-config.yaml** 文件中为 **credentialsMode** 参数设置不同的值，来配置 CCO 来满足机构的安全要求。

如果您不希望在集群 **kube-system** 项目中存储管理员级别的凭证 secret，您可以在安装 OpenShift Container Platform 时把 CCO 的 **credentialsMode** 参数设置为 **Manual**，并手动管理您的云凭证。

使用手动模式可允许每个集群组件只拥有所需的权限，而无需在集群中存储管理员级别的凭证。如果您的环境没有连接到云供应商公共 IAM 端点，您还可以使用此模式。但是，每次升级都必须手动将权限与新发行镜像协调。您还必须手动为每个请求它们的组件提供凭证。

其他资源

- 有关所有可用 CCO 凭证模式及其支持的平台的更多信息，请参阅关于 Cloud Credential Operator。

3.2.2. 手动创建 IAM

在无法访问云身份和访问管理（IAM）API 的环境中，或者管理员更不希望将管理员级别的凭证 secret 存储在集群 **kube-system** 命名空间中时，可以在安装前将 Cloud Credential Operator（CCO）放入手动模式。

流程

1. 要生成清单，请在包含安装程序的目录中运行以下命令：

```
$ openshift-install create manifests --dir <installation_directory> 1
```

1 对于 **<installation_directory>**，请指定用于保存安装程序所创建的文件的目录名称。

2. 在 manifests 目录中插入配置映射，以便 Cloud Credential Operator 放置到手动模式中：

```
$ cat <<EOF > mycluster/manifests/cco-configmap.yaml
apiVersion: v1
kind: ConfigMap
metadata:
  name: cloud-credential-operator-config
  namespace: openshift-cloud-credential-operator
  annotations:
    release.openshift.io/create-only: "true"
data:
  disabled: "true"
EOF
```

3. 删除使用本地云凭证创建的 **admin** 凭证 secret。这会防止您的 **admin** 凭证存储在集群中：

```
$ rm mycluster/openshift/99_cloud-creds-secret.yaml
```

4. 从包含安装程序的目录中，获取 **openshift-install** 二进制文件要使用的 OpenShift Container Platform 发行镜像详情：

```
$ openshift-install version
```

输出示例

```
release image quay.io/openshift-release-dev/ocp-release:4.y.z-x86_64
```

5. 针对您要部署到的云，找到此发行版本镜像中的所有 **CredentialsRequests** 对象：

```
$ oc adm release extract quay.io/openshift-release-dev/ocp-release:4.y.z-x86_64 --
credentials-requests --cloud=azure
```

这会显示每个请求的详情。

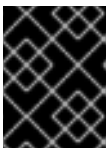
CredentialsRequest 对象示例

```
apiVersion: cloudcredential.openshift.io/v1
kind: CredentialsRequest
metadata:
  labels:
    controller-tools.k8s.io: "1.0"
  name: openshift-image-registry-azure
  namespace: openshift-cloud-credential-operator
spec:
  secretRef:
    name: installer-cloud-credentials
    namespace: openshift-image-registry
  providerSpec:
    apiVersion: cloudcredential.openshift.io/v1
    kind: AzureProviderSpec
    roleBindings:
    - role: Contributor
```

6. 在之前生成的 **openshift-install** 清单目录中为 secret 创建 YAML 文件。secret 必须使用在 **spec.secretRef** 中为每个 **credentialsRequest** 定义的命名空间和 secret 名称存储 。secret 数据的格式因云供应商而异。

7. 从包含安装程序的目录中，开始创建集群：

```
$ openshift-install create cluster --dir <installation_directory>
```



重要

在升级使用手动维护凭证的集群前，必须确保 CCO 处于可升级状态。详情请参阅您的云供应商的*对手动维护凭证的集群进行升级*部分的内容。

3.2.3. 管理凭证 root secret 格式

每个云供应商都使用 **kube-system** 命名空间中的一个凭证 root secret，用于满足所有凭证请求并创建它们相应的 secret。这可以通过 mint 新凭证（*mint mode*），或复制凭证 root secret（*passthrough mode*）实现。

secret 的格式因云而异，也用于每个 **CredentialsRequest** secret。

Microsoft Azure secret 格式

```
apiVersion: v1
kind: Secret
metadata:
  namespace: kube-system
  name: azure-credentials
stringData:
  azure_subscription_id: <SubscriptionID>
  azure_client_id: <ClientID>
  azure_client_secret: <ClientSecret>
  azure_tenant_id: <TenantID>
  azure_resource_prefix: <ResourcePrefix>
  azure_resourcegroup: <ResourceGroup>
  azure_region: <Region>
```

在 Microsoft Azure 中，凭证 secret 格式包括两个必须包含集群基础架构 ID 的属性，每个集群安装随机生成。该值可在运行创建清单后找到：

```
$ cat .openshift_install_state.json | jq '."*installconfig.ClusterID".InfraID' -r
```

输出示例

```
mycluster-2mpcn
```

这个值将在 secret 数据中使用，如下所示：

```
azure_resource_prefix: mycluster-2mpcn
azure_resourcegroup: mycluster-2mpcn-rg
```

3.2.4. 使用手动维护的凭证升级集群

如果在未来的发行版本中添加了凭证，则使用手动维护凭证的集群的 Cloud Credential Operator（CCO）**可升级**状态会变为 **false**。对于次版本（例如从 4.5 到 4.6），这个状态会阻止升级，直到解决了更新的权限。对于 z-stream 版本（例如从 4.5.10 到 4.5.11），升级不会受阻，但必须为新版本更新凭证。

使用 Web 控制台的 **Administrator** 视角来判断 CCO 是否可以升级。

1. 导航至 **Administration → Cluster Settings**。
2. 要查看 CCO 状态详情，请点 **Cluster Operators** 列表中的 **cloud-credential**。
3. 如果 **Conditions** 部分中的 **Upgradeable** 状态为 **False**，请检查新发行版本的 **credentialsRequests**，并在升级前更新集群中手动维护的凭证以匹配。

除了为您要升级到的发行版本镜像创建新凭证外，还需要查看现有凭证所需的权限，并满足新发行版本中现有组件的所有新权限要求。CCO 无法检测到这些不匹配的问题，且在此情况下无法将 **upgradable** 设置为 **false**。

详情请参阅您的云供应商的*手动创建 IAM* 部分来了解如何获取和使用您的云所需的凭证。

3.2.5. Mint 模式

Mint 模式是 OpenShift Container Platform 的默认和推荐的 Cloud Credential Operator（CCO）凭证模式。在这种模式中，CCO 使用提供的管理员级云凭证来运行集群。AWS、GCP 和 Azure 支持 Mint 模式。

在 mint 模式中，**admin** 凭证存储在 **kube-system** 命名空间中，然后由 CCO 使用来处理集群中的 **CredentialsRequest** 对象，并为每个对象创建具有特定权限的用户。

mint 模式的好处包括：

- 每个集群组件只有其所需权限
- 云凭证的自动、持续协调，包括升级可能需要的额外凭证或权限

mint 模式的一个缺陷是，**admin** 凭证需要存储在集群 **kube-system** 的 secret 中。

3.2.6. 后续步骤

- 安装 OpenShift Container Platform 集群：
  - 使用安装程序置备的基础架构默认选项在 Azure 上快速安装集群
  - 在安装程序置备的基础架构中使用云自定义安装集群
  - 使用网络自定义在安装程序置备的基础架构上安装集群

3.3. 在 AZURE 上快速安装集群

在 OpenShift Container Platform 版本 4.6 中，您可以使用默认配置选项在 Microsoft Azure 上安装集群。

3.3.1. 先决条件

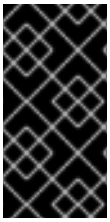
- 查看有关 OpenShift Container Platform 安装和更新流程的详细信息。
- 配置一个 Azure 帐户以托管集群，并决定要将集群部署到的已测试和验证的区域。
- 如果使用防火墙，则必须将其配置为允许集群需要访问的站点。
- 如果不允许系统管理身份和访问管理（IAM），集群管理员可以 手动创建和维护 IAM 凭证。手动模式也可以用于云 IAM API 无法访问的环境中。

3.3.2. OpenShift Container Platform 的互联网访问

在 OpenShift Container Platform 4.6 中，您需要访问互联网来安装集群。

您必须具有以下互联网访问权限：

- 访问 OpenShift Cluster Manager 以下载安装程序并执行订阅管理。如果集群可以访问互联网，并且没有禁用 Telemetry，该服务会自动授权您的集群。
- 访问 Quay.io，以获取安装集群所需的软件包。
- 获取执行集群更新所需的软件包。

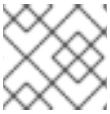


重要

如果您的集群无法直接访问互联网，则可以在置备的某些类基础架构上执行受限网络安装。在此过程中，您要下载所需的内容，并使用它在镜像 registry（mirror registry） 中填充安装集群并生成安装程序所需的软件包。对于某些安装类型，集群要安装到的环境不需要访问互联网。在更新集群之前，要更新 registry 镜像系统中的内容。

3.3.3. 生成 SSH 私钥并将其添加到代理中

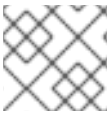
如果要在集群上执行安装调试或灾难恢复，则必须为 **ssh-agent** 和安装程序提供 SSH 密钥。您可以使用此密钥访问公共集群中的 bootstrap 机器来排除安装问题。



注意

在生产环境中，您需要进行灾难恢复和调试。

您可以使用此密钥以 **core** 用户身份通过 SSH 连接到 master 节点。在部署集群时，此密钥会添加到 **core** 用户的 **~/.ssh/authorized_keys** 列表中。



注意

您必须使用一个本地密钥，而不要使用在特定平台上配置的密钥，如 AWS 密钥对。

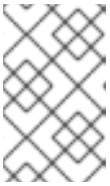
流程

1. 如果还没有为计算机上免密码身份验证而配置的 SSH 密钥，请创建一个。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ ssh-keygen -t ed25519 -N '' \
  -f <path>/<file_name> 1
```

1 指定新 SSH 密钥的路径和文件名，如 **~/.ssh/id_rsa**。如果您已有密钥对，请确保您的公钥位于 **~/.ssh** 目录中。

运行此命令会在指定的位置生成不需要密码的 SSH 密钥。



注意

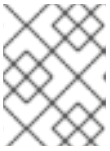
如果您计划在 **x86_64** 架构中安装使用 FIPS 验证的/Modules in Process 加密库的 OpenShift Container Platform 集群，不要创建使用 **ed25519** 算法的密钥。反之，创建一个使用 **rsa** 或 **ecdsa** 算法的密钥。

2. 作为后台任务启动 **ssh-agent** 进程：

```
$ eval "$(ssh-agent -s)"
```

输出示例

```
Agent pid 31874
```



注意

如果您的集群采用 FIPS 模式，则只使用 FIPS 兼容算法来生成 SSH 密钥。密钥必须是 RSA 或 ECDSA。

3. 将 SSH 私钥添加到 **ssh-agent**：

```
$ ssh-add <path>/<file_name> 1
```

输出示例

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

1 指定 SSH 私钥的路径和文件名，如 **~/.ssh/id_rsa**

后续步骤

- 在安装 OpenShift Container Platform 时，为安装程序提供 SSH 公钥。

3.3.4. 获取安装程序

在安装 OpenShift Container Platform 之前，将安装文件下载到本地计算机上。

先决条件

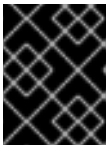
- 运行 Linux 或 macOS 的计算机，本地磁盘空间为 500 MB

流程

1. 访问 OpenShift Cluster Manager 站点的 Infrastructure Provider 页面。如果您有红帽帐号，请使用自己的凭证登录。如果没有，请创建一个帐户。
2. 选择您的基础架构供应商。
3. 进入适用于您的安装类型的页面，下载您的操作系统的安装程序，并将文件放在要保存安装配置文件的目录中。。

重要

安装程序会在用来安装集群的计算机上创建若干文件。在完成集群安装后，您必须保留安装程序和安装程序所创建的文件。这两个文件都需要删除集群。



重要

删除安装程序创建的文件不会删除您的集群，即使集群在安装过程中失败也是如此。要删除集群，为特定云供应商完成 OpenShift Container Platform 卸载流程。

4. 提取安装程序。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ tar xvf openshift-install-linux.tar.gz
```

5. 从 Red Hat OpenShift Cluster Manager 下载安装 pull secret 。通过此 pull secret，您可以进行所含授权机构提供的服务的身份验证，这些服务包括为 OpenShift Container Platform 组件提供容器镜像的 Quay.io。

3.3.5. 部署集群

您可以在兼容云平台中安装 OpenShift Container Platform。

重要

安装程序的 **create cluster** 命令只能在初始安装过程中运行一次。

先决条件

- 配置托管集群的云平台的帐户。
- 获取 OpenShift Container Platform 安装程序以及集群的 pull secret。

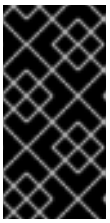
流程

1. 更改为包含安装程序的目录并初始化集群部署：

```
$ ./openshift-install create cluster --dir <installation_directory> \ 1
    --log-level=info 2
```

1 对于 **<installation_directory>**，请指定用于保存安装程序所创建的文件的目录名称。

2 要查看不同的安装详情，请指定 **warn**、**debug** 或 **error**，而不要指定 **info**。



重要

指定一个空目录。一些安装信息，如 bootstrap X.509 证书，有较短的过期间隔，因此不要重复使用安装目录。如果要重复使用另一个集群安装中的个别文件，可以将其复制到您的目录中。但是，一些安装数据的文件名可能会在发行版本之间有所改变。从 OpenShift Container Platform 老版本中复制安装文件时要格外小心。

在提示符处提供值：

a. 可选：选择用来访问集群机器的 SSH 密钥。

注意

对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 **ssh-agent** 进程使用的 SSH 密钥。

b. 选择 **azure** 作为目标平台。

c. 如果计算机上没有 Microsoft Azure 配置集，请为您的订阅和服务主体指定以下 Azure 参数值：

- **azure subscription id**：要用于集群的订阅 ID。指定帐户输出中的 **id** 值。
- **azure tenant id**：租户 ID。指定帐户输出中的 **tenantId** 值。

- **azure service principal client id**：服务主体的 **appId** 参数值。
- **azure service principal client secret**：服务主体的 **password** 参数值。

d. 选择要在其中部署集群的区域。

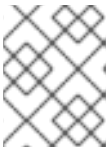
e. 选择集群要部署到的基域。基域与您为集群创建的 Azure DNS 区对应。

f. 为集群输入一个描述性名称。

重要

所有通过公共端点提供的 Azure 资源均存在资源名称的限制，您无法创建使用某些名称的资源。如需 Azure 限制词语列表，请参阅 Azure 文档中的解决保留资源名称错误。

g. 粘贴 Red Hat OpenShift Cluster Manager 中的 pull secret 。



注意

如果您在主机上配置的云供应商帐户没有足够的权限来部署集群，安装过程将会停止，并且显示缺少的权限。

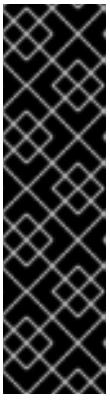
集群部署完成后，终端会显示访问集群的信息，包括指向其 Web 控制台的链接和 **kubeadmin** 用户的凭证。

输出示例

```
...
INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run 'export
KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'
INFO Access the OpenShift web-console here: https://console-openshift-
console.apps.mycluster.example.com
INFO Login to the console with user: "kubeadmin", and password: "4vYBz-Ee6gm-ymBZj-
Wt5AL"
INFO Time elapsed: 36m22s
```

注意

当安装成功时，集群访问和凭证信息还会输出到 **<installation_directory>/.openshift_install.log**。



重要

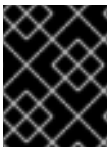
- 安装程序生成的 Ignition 配置文件包含在 24 小时后过期的证书，然后在过期时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外情况是，您需要手动批准待处理的 **node-bootstrapper** 证书签名请求（CSR）来恢复 kubelet 证书。如需更多信息，请参阅*从过期的 control plane 证书中恢复*的文档。
- 建议您在生成 12 小时后使用 Ignition 配置文件，因为集群安装后 24 小时证书从 16 小时轮转至 22 小时。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中运行证书更新时避免安装失败。

重要

您不得删除安装程序或安装程序所创建的文件。需要这两者才能删除集群。

3.3.6. 通过下载二进制文件安装 OpenShift CLI

您需要安装 CLI（**oc**） 来使用命令行界面与 OpenShift Container Platform 进行交互。您可在 Linux 、Windows 或 macOS 上安装 **oc**。



重要

如果安装了旧版本的 **oc**，则无法使用 OpenShift Container Platform 4.6 中的所有命令。下载并安装新版本的 **oc**。

3.3.6.1. 在 Linux 上安装 OpenShift CLI

您可以按照以下流程在 Linux 上安装 OpenShift CLI（**oc**）二进制文件。

流程

1. 进入到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 在 **Version** 下拉菜单中选择相应的版本。
3. 单击 **OpenShift v4.6 Linux 客户端** 条目旁边的 **Download Now**，再保存文件。
4. 解包存档：

```
$ tar xvzf <file>
```

5. 把 **oc** 二进制代码放到 **PATH** 中的目录中。
执行以下命令可以查看当前的 **PATH** 设置：

```
$ echo $PATH
```

安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
$ oc <command>
```

3.3.6.2. 在 Windows 上安装 OpenShift CLI

您可以按照以下流程在 Windows 上安装 OpenShift CLI（**oc**）二进制代码。

流程

1. 进入到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 在 **Version** 下拉菜单中选择相应的版本。
3. 单击 **OpenShift v4.6 Windows 客户端**条目旁边的 **Download Now**，再保存文件。
4. 使用 ZIP 程序解压存档。

5. 把 **oc** 二进制代码放到 **PATH** 中的目录中。
   要查看您的 **PATH**，请打开命令提示窗口并执行以下命令：

```
C:\> path
```

安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
C:\> oc <command>
```

3.3.6.3. 在 macOS 上安装 OpenShift CLI

您可以按照以下流程在 macOS 上安装 OpenShift CLI（**oc**）二进制代码。

流程

1. 进入到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 在 **Version** 下拉菜单中选择相应的版本。
3. 单击 **OpenShift v4.6 MacOSX 客户端**条目旁边的 **Download Now**，再保存文件。
4. 解包和解压存档。
5. 将 **oc** 二进制文件移到 PATH 的目录中。
   要查看您的 **PATH**，打开一个终端窗口并执行以下命令：

```
$ echo $PATH
```

安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
$ oc <command>
```

3.3.7. 使用 CLI 登录到集群

您可以通过导出集群 **kubeconfig** 文件，以默认系统用户身份登录集群。**kubeconfig** 文件包含关于集群的信息，供 CLI 用于将客户端连接到正确集群和 API 服务器。该文件特只适用于一个特定的集群，在 OpenShift Container Platform 安装过程中创建。

先决条件

- 已部署了 OpenShift Container Platform 集群。
- 已安装 **oc** CLI。

流程

1. 导出 **kubeadmin** 凭证：

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

2. 使用导出的配置，验证能否成功运行 **oc** 命令：

```
$ oc whoami
```

输出示例

```
system:admin
```

其他资源

- 如需有关访问和了解 OpenShift Container Platform Web 控制台的更多信息，请参阅访问 Web 控制台。

3.3.8. OpenShift Container Platform 的 Telemetry 访问

在 OpenShift Container Platform 4.6 中，默认运行的 Telemetry 服务提供有关集群健康状况和成功更新的指标，需要访问互联网。如果您的集群连接到互联网，Telemetry 会自动运行，而且集群会注册到 OpenShift Cluster Manager。

确认 OpenShift Cluster Manager 清单正确后，可以由 Telemetry 自动维护，也可以使用 OpenShift Cluster Manager 手动维护，使用订阅监控来跟踪帐户或多集群级别的 OpenShift Container Platform 订阅。

其他资源

- 有关 Telemetry 服务的更多信息，请参阅关于远程健康监控。

3.3.9. 后续步骤

- 自定义集群。
- 若有需要，您可以选择不使用远程健康报告。

3.4. 使用自定义在 AZURE 上安装集群

在 OpenShift Container Platform 版本 4.6 中，您可以在安装程序在 Microsoft Azure 上置备的基础架构上安装自定义的集群。要自定义安装，请在安装集群前修改 **install-config.yaml** 文件中的参数。

3.4.1. 先决条件

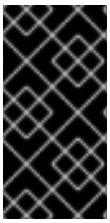
- 查看有关 OpenShift Container Platform 安装和更新流程的详细信息。
- 配置一个 Azure 帐户以托管集群，并决定要将集群部署到的已测试和验证的区域。
- 如果使用防火墙，则必须将其配置为允许集群需要访问的站点。
- 如果不允许系统管理身份和访问管理（IAM），集群管理员可以 手动创建和维护 IAM 凭证。手动模式也可以用于云 IAM API 无法访问的环境中。

3.4.2. OpenShift Container Platform 的互联网访问

在 OpenShift Container Platform 4.6 中，您需要访问互联网来安装集群。

您必须具有以下互联网访问权限：

- 访问 OpenShift Cluster Manager 以下载安装程序并执行订阅管理。如果集群可以访问互联网，并且没有禁用 Telemetry，该服务会自动授权您的集群。
- 访问 Quay.io，以获取安装集群所需的软件包。
- 获取执行集群更新所需的软件包。



重要

如果您的集群无法直接访问互联网，则可以在置备的某些类基础架构上执行受限网络安装。在此过程中，您要下载所需的内容，并使用它在镜像 registry（mirror registry） 中填充安装集群并生成安装程序所需的软件包。对于某些安装类型，集群要安装到的环境不需要访问互联网。在更新集群之前，要更新 registry 镜像系统中的内容。

3.4.3. 生成 SSH 私钥并将其添加到代理中

如果要在集群上执行安装调试或灾难恢复，则必须为 **ssh-agent** 和安装程序提供 SSH 密钥。您可以使用此密钥访问公共集群中的 bootstrap 机器来排除安装问题。

注意

在生产环境中，您需要进行灾难恢复和调试。

您可以使用此密钥以 **core** 用户身份通过 SSH 连接到 master 节点。在部署集群时，此密钥会添加到 **core** 用户的 **~/.ssh/authorized_keys** 列表中。

注意

您必须使用一个本地密钥，而不要使用在特定平台上配置的密钥，如 AWS 密钥对。

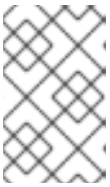
流程

1. 如果还没有为计算机上免密码身份验证而配置的 SSH 密钥，请创建一个。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ ssh-keygen -t ed25519 -N '' \
  -f <path>/<file_name> 1
```

1 指定新 SSH 密钥的路径和文件名，如 **~/.ssh/id_rsa**。如果您已有密钥对，请确保您的公钥位于 **~/.ssh** 目录中。

运行此命令会在指定的位置生成不需要密码的 SSH 密钥。



注意

如果您计划在 **x86_64** 架构中安装使用 FIPS 验证的/Modules in Process 加密库的 OpenShift Container Platform 集群，不要创建使用 **ed25519** 算法的密钥。反之，创建一个使用 **rsa** 或 **ecdsa** 算法的密钥。

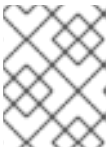
2. 作为后台任务启动 **ssh-agent** 进程：

```
-
```

```
$ eval "$(ssh-agent -s)"
```

输出示例

```
Agent pid 31874
```



注意

如果您的集群采用 FIPS 模式，则只使用 FIPS 兼容算法来生成 SSH 密钥。密钥必须是 RSA 或 ECDSA。

3. 将 SSH 私钥添加到 **ssh-agent**：

```
$ ssh-add <path>/<file_name> 1
```

输出示例

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

1 指定 SSH 私钥的路径和文件名，如 **~/.ssh/id_rsa**

后续步骤

- 在安装 OpenShift Container Platform 时，为安装程序提供 SSH 公钥。

3.4.4. 获取安装程序

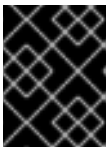
在安装 OpenShift Container Platform 之前，将安装文件下载到本地计算机上。

先决条件

- 运行 Linux 或 macOS 的计算机，本地磁盘空间为 500 MB

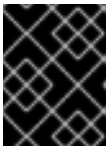
流程

1. 访问 OpenShift Cluster Manager 站点的 Infrastructure Provider 页面。如果您有红帽帐号，请使用自己的凭证登录。如果没有，请创建一个帐户。
2. 选择您的基础架构供应商。
3. 进入适用于您的安装类型的页面，下载您的操作系统的安装程序，并将文件放在要保存安装配置文件的目录中。。



重要

安装程序会在用来安装集群的计算机上创建若干文件。在完成集群安装后，您必须保留安装程序和安装程序所创建的文件。这两个文件都需要删除集群。



重要

删除安装程序创建的文件不会删除您的集群，即使集群在安装过程中失败也是如此。要删除集群，为特定云供应商完成 OpenShift Container Platform 卸载流程。

4. 提取安装程序。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ tar xvf openshift-install-linux.tar.gz
```

5. 从 Red Hat OpenShift Cluster Manager 下载安装 pull secret 。通过此 pull secret，您可以进行所含授权机构提供的服务的身份验证，这些服务包括为 OpenShift Container Platform 组件提供容器镜像的 Quay.io。

3.4.5. 创建安装配置文件

您可以自定义在 Microsoft Azure 上安装的 OpenShift Container Platform 集群。

先决条件

- 获取 OpenShift Container Platform 安装程序以及集群的 pull secret。

流程

1. 创建 **install-config.yaml** 文件。
   a. 更改到包含安装程序的目录，再运行以下命令：

```
$ ./openshift-install create install-config --dir <installation_directory> 1
```

   1 对于 **<installation_directory>**，请指定用于保存安装程序所创建的文件的目录名称。

重要

指定一个空目录。一些安装信息，如 bootstrap X.509 证书，有较短的过期间隔，因此不要重复使用安装目录。如果要重复使用另一个集群安装中的个别文件，可以将其复制到您的目录中。但是，一些安装数据的文件名可能会在发行版本之间有所改变。从 OpenShift Container Platform 老版本中复制安装文件时要格外小心。

   b. 在提示符处，提供您的云的配置详情：
      i. 可选：选择用来访问集群机器的 SSH 密钥。

注意

对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 **ssh-agent** 进程使用的 SSH 密钥。

      ii. 选择 **azure** 作为目标平台。
      iii. 如果计算机上没有 Microsoft Azure 配置集，请为您的订阅和服务主体指定以下 Azure 参数值：

- azure subscription id：要用于集群的订阅 ID。指定帐户输出中的 **id** 值。
- azure tenant id：租户 ID。指定帐户输出中的 **tenantId** 值。
- azure service principal client id：服务主体的 **appId** 参数值。
- azure service principal client secret：服务主体的 **password** 参数值。

iv. 选择要在其中部署集群的区域。

v. 选择集群要部署到的基域。基域与您为集群创建的 Azure DNS 区对应。

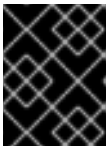
vi. 为集群输入一个描述性名称。

重要

所有通过公共端点提供的 Azure 资源均存在资源名称的限制，您无法创建使用某些名称的资源。如需 Azure 限制词语列表，请参阅 Azure 文档中的解决保留资源名称错误。

vii. 粘贴 Red Hat OpenShift Cluster Manager 中的 pull secret 。

2. 修改 **install-config.yaml** 文件。您可以在**安装配置参数**部分中找到有关可用参数的更多信息。
3. 备份 **install-config.yaml** 文件，以便用于安装多个集群。



重要

install-config.yaml 文件会在安装过程中消耗掉。如果要重复使用此文件，必须现在备份。

3.4.5.1. 安装配置参数

在部署 OpenShift Container Platform 集群前，您可以提供参数值，以描述托管集群的云平台的帐户并选择性地自定义集群平台。在创建 **install-config.yaml** 安装配置文件时，您可以通过命令行来提供所需的参数的值。如果要自定义集群，可以修改 **install-config.yaml** 文件来提供关于平台的更多信息。

注意

安装之后，您无法修改 **install-config.yaml** 文件中的这些参数。

重要

openshift-install 命令不验证参数的字段名称。如果指定了不正确的名称，则不会创建相关的文件或对象，且不会报告错误。确保所有指定的参数的字段名称都正确。

3.4.5.1.1. 所需的配置参数

下表描述了所需的安装配置参数：

表 3.1. 所需的参数

参数	描述	值
apiVersion	**install-config.yaml** 内容的 API 版本。当前版本是 **v1**。安装程序还可能支持旧的 API 版本。	字符串
baseDomain	云供应商的基域。此基础域用于创建到 OpenShift Container Platform 集群组件的路由。集群的完整 DNS 名称是 **baseDomain** 和 **metadata.name** 参数值的组合，其格式为 **<metadata.name>.<baseDomain>**。	完全限定域名或子域名，如 **example.com**。
metadata	Kubernetes 资源 **ObjectMeta**，其中只消耗 **name** 参数。	对象
metadata.name	集群的名称。集群的 DNS 记录是 **{{.metadata.name}}.{{.baseDomain}}** 的子域。	小写字母,连字符(-)和句点(.)的字符串，如 **dev**。
platform	执行安装的具体平台配置：**aws**、**baremetal**、**azure**、**openstack**、**ovirt**、**vsphere**。有关 **platform.<platform>** 参数的额外信息，请参考下表来了解您的具体平台。	对象

pullSecret	从 Red Hat OpenShift Cluster Manager 获取 pull secret，验证从 Quay.io 等服务中下载 OpenShift Container Platform 组件的容器镜像。	{
		"auths":{
		"cloud.openshift.com":{
		"auth":"b3Blb=",
		"email":"you@example.com"
		},
		"quay.io":{
		"auth":"b3Blb=",
		"email":"you@example.com"
		}
		}
		}

3.4.5.1.2. 网络配置参数

您可以根据现有网络基础架构的要求自定义安装配置。例如，您可以扩展集群网络的 IP 地址块，或者提供不同于默认值的不同 IP 地址块。

只支持 IPv4 地址。

表 3.2. 网络参数

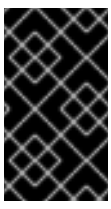
参数	描述	值
networking	集群网络的配置。	对象 **注意** 您不能在安装后修改 **networking** 对象指定的参数。
networking.networkType	要安装的集群网络供应商 Container Network Interface（CNI）插件。	**OpenShiftSDN** 或 **OVNKubernetes**。默认值为 **OpenShiftSDN**。
networking.clusterNetwork	pod 的 IP 地址块。 默认值为 **10.128.0.0/14**，主机前缀为 **/23**。 如果您指定多个 IP 地址块，则块不得互相重叠。	一个对象数组。例如： networking: clusterNetwork: - cidr: 10.128.0.0/14 hostPrefix: 23
networking.clusterNetwork.cidr	使用 **networking.clusterNetwork** 时需要此项。IP 地址块。 一个 IPv4 网络。	使用 CIDR 形式的 IP 地址块。IPv4 块的前缀长度介于 **0** 到 **32** 之间。
networking.clusterNetwork.hostPrefix	分配给每个单独节点的子网前缀长度。例如，如果 **hostPrefix** 设为 **23**，则每个节点从所给的 **cidr** 中分配一个 **/23** 子网。**hostPrefix** 值 **23** 提供 510（2^(32 - 23)- 2）个 pod IP 地址。	子网前缀。 默认值为 **23**。
networking.serviceNetwork	服务的 IP 地址块。默认值为 **172.30.0.0/16**。 OpenShift SDN 和 OVN-Kubernetes 网络供应商只支持服务网络的一个 IP 地址块。	CIDR 格式具有 IP 地址块的数组。例如： networking: serviceNetwork: - 172.30.0.0/16
networking.machineNetwork	机器的 IP 地址块。 如果您指定多个 IP 地址块，则块不得互相重叠。	一个对象数组。例如： networking: machineNetwork: - cidr: 10.0.0.0/16

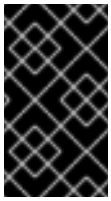
参数	描述	值
networking.machineNetwork.cidr	使用 **networking.machineNetwork** 时需要。IP 地址块。libvirt 以外的所有平台的默认值为 **10.0.0.0/16**。对于 libvirt，默认值为 **192.168.126.0/24**。	CIDR 表示法中的 IP 网络块。 例如： **10.0.0.0/16**。  **注意** 将 **networking.machineNetwork** 设置为与首选 NIC 所在的 CIDR 匹配。

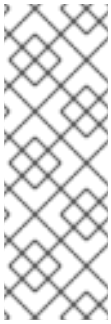
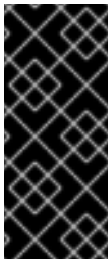
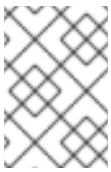
3.4.5.1.3. 可选配置参数

下表描述了可选安装配置参数：

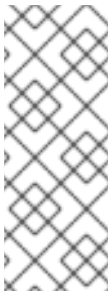
表 3.3. 可选参数

参数	描述	值
additionalTrustBundle	添加到节点可信证书存储中的 PEM 编码 X.509 证书捆绑包。配置了代理时，也可以使用这个信任捆绑包。	字符串
compute	组成计算节点的机器的配置。	machine-pool 对象的数组。详情请查看以下"Machine-pool"表。
compute.architecture	决定池中机器的指令集合架构。目前不支持异构集群，因此所有池都必须指定相同的架构。有效值为 **amd64** （默认值）。	字符串
compute.hyperthreading	是否在计算机器上启用或禁用并发多线程或**超线程**。默认情况下，启用并发多线程以提高机器内核的性能。  **重要** 如果禁用并发多线程，请确保在容量规划时考虑到机器性能可能会显著降低的问题。	**Enabled** 或 **Disabled**
compute.name	使用 **compute** 时需要此值。机器池的名称。	**worker**

参数	描述	值
compute.platform	使用 **compute** 时需要此值。使用此参数指定托管 worker 机器的云供应商。此参数值必须与 **controlPlane.platform** 参数值匹配。	**aws**、**azure**、**gcp**、**openstack**、**ovirt**、**vsphere** 或 **{}**
compute.replicas	要置备的计算机器数量，也称为 worker 机器。	大于或等于 **2** 的正整数。默认值为 **3**。
controlPlane	组成 control plane 的机器的配置。	**MachinePool** 对象的数组。详情请查看以下"Machine-pool"表。
controlPlane.architecture	决定池中机器的指令集合架构。目前不支持异构集群，因此所有池都必须指定相同的架构。有效值为 **amd64** （默认值）。	字符串
controlPlane.hyperthreading	是否在 control plane 机器上启用或禁用并发多线程或**超线程**。默认情况下，启用并发多线程以提高机器内核的性能。  **重要** 如果禁用并发多线程，请确保在容量规划时考虑到机器性能可能会显著降低的问题。	**Enabled** 或 **Disabled**
controlPlane.name	使用 **controlPlane** 时需要。机器池的名称。	**master**
controlPlane.platform	使用 **controlPlane** 时需要。使用此参数指定托管 control plane 机器的云供应商。此参数值必须与 **compute.platform** 参数值匹配。	**aws**、**azure**、**gcp**、**openstack**、**ovirt**、**vsphere** 或 **{}**
controlPlane.replicas	要置备的 control plane 机器数量。	唯一支持的值是 **3**，它是默认值。

参数	描述	值
credentialsMode	Cloud Credential Operator（CCO）模式。如果没有指定任何模式，CCO 会动态地尝试决定提供的凭证的功能，在支持多个模式的平台上使用 mint 模式。  **注意** 不是所有 CCO 模式都支持所有云供应商。如需有关 CCO 模式的更多信息，请参阅*Red Hat Operator 参考指南*内容中的 *Cloud Credential Operator* 条目。	**Mint**、**Passthrough**、**Manual** 或空字符串(**""**)。
fips	启用或禁用 FIPS 模式。默认为 **false**（禁用）。如果启用了 FIPS 模式，运行 OpenShift Container Platform 的 Red Hat Enterprise Linux CoreOS (RHCOS) 机器会绕过默认的 Kubernetes 加密套件，并使用由 RHCOS 提供的加密模块。  **重要** 只有在 **x86_64** 架构中的 OpenShift Container Platform 部署支持 FIPS 验证的/Modules in Process 加密库。  **注意** 如果使用 Azure File 存储，则无法启用 FIPS 模式。	**false** 或 **true**
imageContentSources	release-image 内容的源和仓库。	对象数组。包括一个 **source** 以及可选的 **mirrors**，如下表所示。
imageContentSources.source	使用 **imageContentSources** 时需要。指定用户在镜像拉取规格中引用的仓库。	字符串
imageContentSources.mirrors	指定可能还包含同一镜像的一个或多个仓库。	字符串数组

参数	描述	值
publish	如何发布或公开集群的面向用户的端点，如 Kubernetes API、OpenShift 路由。	**Internal** 或 **External**。把 **publish** 设置为 **Internal** 以部署一个私有集群，它不能被互联网访问。默认值为 **External**。

sshKey	用于验证集群机器访问的 SSH 密钥或密钥。
	 
	注意
	对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 **ssh-agent** 进程使用的 SSH 密钥。
		一个或多个密钥。例如：
		`sshKey:`
		`  <key1>`
		`  <key2>`
		`  <key3>`

3.4.5.1.4. 其他 Azure 配置参数

下表描述了其他 Azure 配置参数：

表 3.4. 其他 Azure 参数

参数	描述	值
compute.platform.azure.osDisk.diskSizeGB	虚拟机的 Azure 磁盘大小。	以 GB 为单位表示磁盘大小的整数。默认值为 **128**。
compute.platform.azure.osDisk.diskType	定义磁盘的类型。	**Standard_LRS**、**Premium_LRS** 或 **标准SSD_LRS**。默认值为 **Premium_LRS**。
controlPlane.platform.azure.osDisk.diskSizeGB	虚拟机的 Azure 磁盘大小。	以 GB 为单位表示磁盘大小的整数。默认值为 **1024**。
controlPlane.platform.azure.osDisk.diskType	定义磁盘的类型。	**Premium_LRS** 或 **标准SSD_LRS**. 默认值为 **Premium_LRS**。
platform.azure.baseDomainResourceGroupName	包含基域的 DNS 区的资源组的名称。	字符串，如 **production_cluster**。

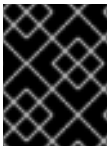
参数	描述	值
platform.azure.outboundType	用于将集群连接到互联网的出站路由策略。如果使用用户定义的路由，则必须在安装集群前配置出站路由。安装程序不负责配置用户定义的路由。	**loadbalancer** 或 **UserDefinedRouting**。默认为 **LoadBalancer**。
platform.azure.region	托管集群的 Azure 区域名称。	任何有效的区域名称，如 **centralus**。
platform.azure.zone	可在其中放入机器的可用区的列表。如需高可用性，请至少指定两个区域。	区域列表，如 **["1", "2", "3"]**。
platform.azure.networkResourceGroupName	包含要将集群部署到的现有 VNet 的资源组名称。这个名称不能和 **platform.azure.baseDomainResourceGroupName**相同。	字符串。
platform.azure.virtualNetwork	要将集群部署到的现有 VNet 的名称。	字符串。
platform.azure.controlPlaneSubnet	要将 control plane 机器部署到的 VNet 中现有子网的名称。	有效的 CIDR，如 **10.0.0.0/16**。
platform.azure.computeSubnet	您要将计算机器部署到的 VNet 中现有子网的名称。	有效的 CIDR，如 **10.0.0.0/16**。
platform.azure.cloudName	用于使用适当 Azure API 端点配置 Azure SDK 的 Azure 云环境名称。如果为空，则使用默认值 **AzurePublicCloud**。	任何有效的云环境，如 **AzurePublicCloud** 或 **AzureUSGovernmentCloud**。

注意

您无法自定义 Azure 可用区，也不能使用标签来整理用于 Azure 集群的 Azure 资源。

3.4.5.2. Azure 的自定义 install-config.yaml 文件示例

您可以自定义 **install-config.yaml** 文件，以指定有关 OpenShift Container Platform 集群平台的更多信息，或修改所需参数的值。



重要

此示例 YAML 文件仅供参考。您必须使用安装程序来获取 **install-config.yaml** 文件，并且修改该文件。

```
apiVersion: v1
baseDomain: example.com (1)
controlPlane: (2)
  hyperthreading: Enabled (3) (4)
  name: master
```

```
    platform:
      azure:
        osDisk:
          diskSizeGB: 1024 5
          diskType: Premium_LRS
        type: Standard_D8s_v3
    replicas: 3
  compute: 6
  - hyperthreading: Enabled 7
    name: worker
    platform:
      azure:
        type: Standard_D2s_v3
        osDisk:
          diskSizeGB: 512 8
          diskType: Standard_LRS
        zones: 9
        - "1"
        - "2"
        - "3"
    replicas: 5
  metadata:
    name: test-cluster 10
  networking:
    clusterNetwork:
    - cidr: 10.128.0.0/14
      hostPrefix: 23
    machineNetwork:
    - cidr: 10.0.0.0/16
    networkType: OpenShiftSDN
    serviceNetwork:
    - 172.30.0.0/16
  platform:
    azure:
      baseDomainResourceGroupName: resource_group 11
      region: centralus 12
      resourceGroupName: existing_resource_group 13
      outboundType: Loadbalancer
      cloudName: AzurePublicCloud
  pullSecret: '{"auths": ...}' 14
  fips: false 15
  sshKey: ssh-ed25519 AAAA... 16
```

1 10 12 14 必需。安装程序会提示您输入这个值。

2 6 如果没有提供这些参数和值，安装程序会提供默认值。

3 7 **controlPlane** 部分是一个单映射，但 **compute** 部分是一系列映射。为满足不同数据结构的要求，**compute** 部分的第一行必须以连字符 **-** 开头，**controlPlane** 部分的第一行则不可以连字符开头。只使用一个 control plane 池。

4 是否要启用或禁用并发多线程或**超线程**。默认情况下，启用并发多线程以提高机器内核的性能。您可以通过将参数值设为 **Disabled** 来禁用。如果您在某些集群机器上禁用并发多线程，则必须在所有集群机器上禁用。

重要

如果禁用并发多线程，请确保在容量规划时考虑到机器性能可能会显著降低的问题。如果禁用并发多线程，请使用较大的虚拟机类型，如 **Standard_D8s_v3**。

5 8 可以 GB 为单位指定要使用的磁盘大小。control plane 节点（也称为 master 节点）的最低推荐值为 1024 GB。

9 指定要将机器部署到的区域列表。如需高可用性，请至少指定两个区域。

11 指定包含基域的 DNS 区的资源组的名称。

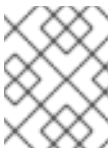
13 指定要安装集群的现有资源组的名称。如果未定义，则会为集群创建新的资源组。

15 是否启用或禁用 FIPS 模式。默认情况下不启用 FIPS 模式。如果启用了 FIPS 模式，运行 OpenShift Container Platform 的 Red Hat Enterprise Linux CoreOS (RHCOS) 机器会绕过默认的 Kubernetes 加密套件，并使用由 RHCOS 提供的加密模块。

重要

只有在 **x86_64** 架构中的 OpenShift Container Platform 部署支持 FIPS 验证的/Modules in Process 加密库。

16 您可以选择提供您用来访问集群中机器的 **sshKey** 值。



注意

对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 **ssh-agent** 进程使用的 SSH 密钥。

3.4.5.3. 在安装过程中配置集群范围代理

生产环境可能会拒绝直接访问互联网，而是提供 HTTP 或 HTTPS 代理。您可以通过在 **install-config.yaml** 文件中配置代理设置，将新的 OpenShift Container Platform 集群配置为使用代理。

先决条件

- 您有一个现有的 **install-config.yaml** 文件。
- 您检查了集群需要访问的站点，并决定是否需要绕过代理。默认情况下代理所有集群出口流量，包括对托管云供应商 API 的调用。您需要将站点添加到 **Proxy** 对象的 **spec.noProxy** 字段来绕过代理。

注意

Proxy 对象 **status.noProxy** 字段使用安装配置中的 **networking.machineNetwork[].cidr**、**networking.clusterNetwork[].cidr** 和 **networking.serviceNetwork[]** 字段的值填充。

对于在 Amazon Web Services(AWS)、Google Cloud Platform(GCP)、Microsoft Azure 和 Red Hat OpenStack Platform(RHOSP)上安装，**Proxy** 对象 **status.noProxy** 字段也会使用实例元数据端点填充(**169.254.169.254**)。

流程

1. 编辑 **install-config.yaml** 文件并添加代理设置。例如：

```
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: example.com 3
additionalTrustBundle: | 4
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
...
```

1. 用于创建集群外 HTTP 连接的代理 URL。URL 必须是 **http**。
2. 用于创建集群外 HTTPS 连接的代理 URL。
3. 要排除在代理中的目标域名、IP 地址或其他网络 CIDR 的逗号分隔列表。在域前面加 **.** 来仅匹配子域。例如： **.y.com** 匹配 **x.y.com**，但不匹配 **y.com**。使用 * 绕过所有目的地的代理。
4. 如果提供，安装程序会在 **openshift-config 命名空间中生成名为 user-ca- bundle** 的配置映射来保存额外的 CA 证书。如果您提供 **additionalTrustBundle** 和至少一个代理设置，则 **Proxy** 对象会被配置为引用 **trustedCA** 字段中的 **user-ca-bundle** 配置映射。然后，Cluster Network Operator 会创建一个 **trusted-ca-bundle** 配置映射，该配置映射将为 **trustedCA** 参数指定的内容与 RHCOS 信任捆绑包合并。 **additionalTrustBundle** 字段是必需的，除非代理的身份证书由来自 RHCOS 信任捆绑包的颁发机构签名。

注意

安装程序不支持代理的 **readinessEndpoints** 字段。

2. 保存该文件，并在安装 OpenShift Container Platform 时引用。

安装程序会创建一个名为 **cluster** 的集群范围代理，该代理使用提供的 **install-config.yaml** 文件中的代理设置。如果没有提供代理设置，仍然会创建一个 **cluster Proxy** 对象，但它会有一个空 **spec**。

注意

只支持名为 **cluster** 的 **Proxy** 对象，且无法创建额外的代理。

3.4.6. 部署集群

您可以在兼容云平台中安装 OpenShift Container Platform。

重要

安装程序的 **create cluster** 命令只能在初始安装过程中运行一次。

先决条件

- 配置托管集群的云平台的帐户。
- 获取 OpenShift Container Platform 安装程序以及集群的 pull secret。

流程

1. 更改为包含安装程序的目录并初始化集群部署：

```
$ ./openshift-install create cluster --dir <installation_directory> \ 1
    --log-level=info 2
```

1 对于 **<installation_directory>**，请指定自定义 **./install-config.yaml** 文件的位置。

2 要查看不同的安装详情，请指定 **warn**、**debug** 或 **error**，而不要指定 **info**。

注意

如果您在主机上配置的云供应商帐户没有足够的权限来部署集群，安装过程将会停止，并且显示缺少的权限。

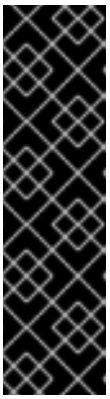
集群部署完成后，终端会显示访问集群的信息，包括指向其 Web 控制台的链接和 **kubeadmin** 用户的凭证。

输出示例

```
...
INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run 'export
KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'
INFO Access the OpenShift web-console here: https://console-openshift-
console.apps.mycluster.example.com
INFO Login to the console with user: "kubeadmin", and password: "4vYBz-Ee6gm-ymBZj-
Wt5AL"
INFO Time elapsed: 36m22s
```

注意

当安装成功时，集群访问和凭证信息还会输出到 **<installation_directory>/.openshift_install.log**。



重要

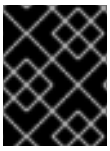
- 安装程序生成的 Ignition 配置文件包含在 24 小时后过期的证书，然后在过期时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外情况是，您需要手动批准待处理的 **node-bootstrapper** 证书签名请求（CSR）来恢复 kubelet 证书。如需更多信息，请参阅*从过期的 control plane 证书中恢复*的文档。
- 建议您在生成 12 小时后使用 Ignition 配置文件，因为集群安装后 24 小时证书从 16 小时轮转至 22 小时。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中运行证书更新时避免安装失败。

重要

您不得删除安装程序或安装程序所创建的文件。需要这两者才能删除集群。

3.4.7. 通过下载二进制文件安装 OpenShift CLI

您需要安装 CLI（**oc**） 来使用命令行界面与 OpenShift Container Platform 进行交互。您可在 Linux 、Windows 或 macOS 上安装 **oc**。



重要

如果安装了旧版本的 **oc**，则无法使用 OpenShift Container Platform 4.6 中的所有命令。下载并安装新版本的 **oc**。

3.4.7.1. 在 Linux 上安装 OpenShift CLI

您可以按照以下流程在 Linux 上安装 OpenShift CLI（**oc**）二进制文件。

流程

1. 进入到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 在 **Version** 下拉菜单中选择相应的版本。
3. 单击 **OpenShift v4.6 Linux 客户端** 条目旁边的 **Download Now**，再保存文件。
4. 解包存档：

```
$ tar xvzf <file>
```

5. 把 **oc** 二进制代码放到 **PATH** 中的目录中。
   执行以下命令可以查看当前的 **PATH** 设置：

```
$ echo $PATH
```

安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
$ oc <command>
```

3.4.7.2. 在 Windows 上安装 OpenShift CLI

您可以按照以下流程在 Windows 上安装 OpenShift CLI（**oc**）二进制代码。

流程

1. 进入到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 在 **Version** 下拉菜单中选择相应的版本。
3. 单击 **OpenShift v4.6 Windows 客户端**条目旁边的 **Download Now**，再保存文件。
4. 使用 ZIP 程序解压存档。

5. 把 **oc** 二进制代码放到 **PATH** 中的目录中。
   要查看您的 **PATH**，请打开命令提示窗口并执行以下命令：

```
C:\> path
```

安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
C:\> oc <command>
```

3.4.7.3. 在 macOS 上安装 OpenShift CLI

您可以按照以下流程在 macOS 上安装 OpenShift CLI（**oc**）二进制代码。

流程

1. 进入到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 在 **Version** 下拉菜单中选择相应的版本。
3. 单击 **OpenShift v4.6 MacOSX 客户端**条目旁边的 **Download Now**，再保存文件。
4. 解包和解压存档。
5. 将 **oc** 二进制文件移到 PATH 的目录中。
   要查看您的 **PATH**，打开一个终端窗口并执行以下命令：

```
$ echo $PATH
```

安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
$ oc <command>
```

3.4.8. 使用 CLI 登录到集群

您可以通过导出集群 **kubeconfig** 文件，以默认系统用户身份登录集群。**kubeconfig** 文件包含关于集群的信息，供 CLI 用于将客户端连接到正确集群和 API 服务器。该文件特只适用于一个特定的集群，在 OpenShift Container Platform 安装过程中创建。

先决条件

- 已部署了 OpenShift Container Platform 集群。
- 已安装 **oc** CLI。

流程

1. 导出 **kubeadmin** 凭证：

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

2. 使用导出的配置，验证能否成功运行 **oc** 命令：

```
$ oc whoami
```

输出示例

```
system:admin
```

其他资源

- 如需有关访问和了解 OpenShift Container Platform Web 控制台的更多信息，请参阅访问 Web 控制台。

3.4.9. OpenShift Container Platform 的 Telemetry 访问

在 OpenShift Container Platform 4.6 中，默认运行的 Telemetry 服务提供有关集群健康状况和成功更新的指标，需要访问互联网。如果您的集群连接到互联网，Telemetry 会自动运行，而且集群会注册到 OpenShift Cluster Manager。

确认 OpenShift Cluster Manager 清单正确后，可以由 Telemetry 自动维护，也可以使用 OpenShift Cluster Manager 手动维护，使用订阅监控来跟踪帐户或多集群级别的 OpenShift Container Platform 订阅。

其他资源

- 有关 Telemetry 服务的更多信息，请参阅关于远程健康监控。

3.4.10. 后续步骤

- 自定义集群。
- 若有需要，您可以选择不使用远程健康报告。

3.5. 使用网络自定义在 AZURE 上安装集群

在 OpenShift Container Platform 版本 4.6 中，您可以使用自定义的网络配置在安装程序在 Microsoft Azure 上置备的基础架构上安装集群。通过自定义网络配置，您的集群可以与环境中现有的 IP 地址分配共存，并与现有的 MTU 和 VXLAN 配置集成。

大部分网络配置参数必须在安装过程中设置，只有 **kubeProxy** 配置参数可以在运行的集群中修改。

3.5.1. 先决条件

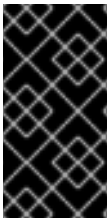
- 查看有关 OpenShift Container Platform 安装和更新流程的详细信息。
- 配置一个 Azure 帐户以托管集群，并决定要将集群部署到的已测试和验证的区域。
- 如果使用防火墙，则必须将其配置为允许集群需要访问的站点。
- 如果不允许系统管理身份和访问管理（IAM），集群管理员可以 手动创建和维护 IAM 凭证。手动模式也可以用于云 IAM API 无法访问的环境中。

3.5.2. OpenShift Container Platform 的互联网访问

在 OpenShift Container Platform 4.6 中，您需要访问互联网来安装集群。

您必须具有以下互联网访问权限：

- 访问 OpenShift Cluster Manager 以下载安装程序并执行订阅管理。如果集群可以访问互联网，并且没有禁用 Telemetry，该服务会自动授权您的集群。
- 访问 Quay.io，以获取安装集群所需的软件包。
- 获取执行集群更新所需的软件包。



重要

如果您的集群无法直接访问互联网，则可以在置备的某些类基础架构上执行受限网络安装。在此过程中，您要下载所需的内容，并使用它在镜像 registry（mirror registry） 中填充安装集群并生成安装程序所需的软件包。对于某些安装类型，集群要安装到的环境不需要访问互联网。在更新集群之前，要更新 registry 镜像系统中的内容。

3.5.3. 生成 SSH 私钥并将其添加到代理中

如果要在集群上执行安装调试或灾难恢复，则必须为 **ssh-agent** 和安装程序提供 SSH 密钥。您可以使用此密钥访问公共集群中的 bootstrap 机器来排除安装问题。

注意

在生产环境中，您需要进行灾难恢复和调试。

您可以使用此密钥以 **core** 用户身份通过 SSH 连接到 master 节点。在部署集群时，此密钥会添加到 **core** 用户的 **~/.ssh/authorized_keys** 列表中。

注意

您必须使用一个本地密钥，而不要使用在特定平台上配置的密钥，如 AWS 密钥对。

流程

1. 如果还没有为计算机上免密码身份验证而配置的 SSH 密钥，请创建一个。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ ssh-keygen -t ed25519 -N '' \
  -f <path>/<file_name> 1
```

1 指定新 SSH 密钥的路径和文件名，如 **~/.ssh/id_rsa**。如果您已有密钥对，请确保您的公钥位于 **~/.ssh** 目录中。

运行此命令会在指定的位置生成不需要密码的 SSH 密钥。

注意

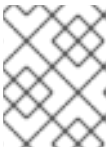
如果您计划在 **x86_64** 架构中安装使用 FIPS 验证的/Modules in Process 加密库的 OpenShift Container Platform 集群，不要创建使用 **ed25519** 算法的密钥。反之，创建一个使用 **rsa** 或 **ecdsa** 算法的密钥。

2. 作为后台任务启动 **ssh-agent** 进程：

```
$ eval "$(ssh-agent -s)"
```

输出示例

```
Agent pid 31874
```



注意

如果您的集群采用 FIPS 模式，则只使用 FIPS 兼容算法来生成 SSH 密钥。密钥必须是 RSA 或 ECDSA。

3. 将 SSH 私钥添加到 **ssh-agent**：

```
$ ssh-add <path>/<file_name> 1
```

输出示例

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

1 指定 SSH 私钥的路径和文件名，如 **~/.ssh/id_rsa**

后续步骤

- 在安装 OpenShift Container Platform 时，为安装程序提供 SSH 公钥。

3.5.4. 获取安装程序

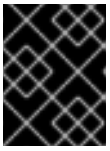
在安装 OpenShift Container Platform 之前，将安装文件下载到本地计算机上。

先决条件

- 运行 Linux 或 macOS 的计算机，本地磁盘空间为 500 MB

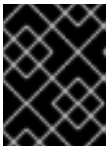
流程

1. 访问 OpenShift Cluster Manager 站点的 Infrastructure Provider 页面。如果您有红帽帐号，请使用自己的凭证登录。如果没有，请创建一个帐户。
2. 选择您的基础架构供应商。
3. 进入适用于您的安装类型的页面，下载您的操作系统的安装程序，并将文件放在要保存安装配置文件的目录中。。



重要

安装程序会在用来安装集群的计算机上创建若干文件。在完成集群安装后，您必须保留安装程序和安装程序所创建的文件。这两个文件都需要删除集群。



重要

删除安装程序创建的文件不会删除您的集群，即使集群在安装过程中失败也是如此。要删除集群，为特定云供应商完成 OpenShift Container Platform 卸载流程。

4. 提取安装程序。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ tar xvf openshift-install-linux.tar.gz
```

5. 从 Red Hat OpenShift Cluster Manager 下载安装 pull secret 。通过此 pull secret，您可以进行所含授权机构提供的服务的身份验证，这些服务包括为 OpenShift Container Platform 组件提供容器镜像的 Quay.io。

3.5.5. 创建安装配置文件

您可以自定义在 Microsoft Azure 上安装的 OpenShift Container Platform 集群。

先决条件

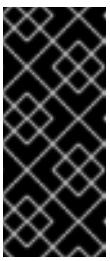
- 获取 OpenShift Container Platform 安装程序以及集群的 pull secret。

流程

1. 创建 **install-config.yaml** 文件。
   a. 更改到包含安装程序的目录，再运行以下命令：

```
$ ./openshift-install create install-config --dir <installation_directory> 1
```

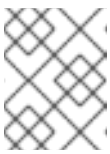
   1 对于 **<installation_directory>**，请指定用于保存安装程序所创建的文件的目录名称。



重要

指定一个空目录。一些安装信息，如 bootstrap X.509 证书，有较短的过期间隔，因此不要重复使用安装目录。如果要重复使用另一个集群安装中的个别文件，可以将其复制到您的目录中。但是，一些安装数据的文件名可能会在发行版本之间有所改变。从 OpenShift Container Platform 老版本中复制安装文件时要格外小心。

   b. 在提示符处，提供您的云的配置详情：
      i. 可选：选择用来访问集群机器的 SSH 密钥。



注意

对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 **ssh-agent** 进程使用的 SSH 密钥。

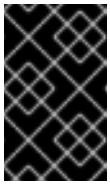
      ii. 选择 **azure** 作为目标平台。
      iii. 如果计算机上没有 Microsoft Azure 配置集，请为您的订阅和服务主体指定以下 Azure 参数值：

- azure subscription id：要用于集群的订阅 ID。指定帐户输出中的 **id** 值。
- azure tenant id：租户 ID。指定帐户输出中的 **tenantId** 值。
- azure service principal client id：服务主体的 **appId** 参数值。
- azure service principal client secret：服务主体的 **password** 参数值。

iv. 选择要在其中部署集群的区域。

v. 选择集群要部署到的基域。基域与您为集群创建的 Azure DNS 区对应。

vi. 为集群输入一个描述性名称。

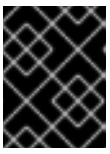


重要

所有通过公共端点提供的 Azure 资源均存在资源名称的限制，您无法创建使用某些名称的资源。如需 Azure 限制词语列表，请参阅 Azure 文档中的解决保留资源名称错误。

vii. 粘贴 Red Hat OpenShift Cluster Manager 中的 pull secret 。

2. 修改 **install-config.yaml** 文件。您可以在**安装配置参数**部分中找到有关可用参数的更多信息。
3. 备份 **install-config.yaml** 文件，以便用于安装多个集群。



重要

install-config.yaml 文件会在安装过程中消耗掉。如果要重复使用此文件，必须现在备份。

3.5.5.1. 安装配置参数

在部署 OpenShift Container Platform 集群前，您可以提供参数值，以描述托管集群的云平台的帐户并选择性地自定义集群平台。在创建 **install-config.yaml** 安装配置文件时，您可以通过命令行来提供所需的参数的值。如果要自定义集群，可以修改 **install-config.yaml** 文件来提供关于平台的更多信息。

注意

安装之后，您无法修改 **install-config.yaml** 文件中的这些参数。

重要

openshift-install 命令不验证参数的字段名称。如果指定了不正确的名称，则不会创建相关的文件或对象，且不会报告错误。确保所有指定的参数的字段名称都正确。

3.5.5.1.1. 所需的配置参数

下表描述了所需的安装配置参数：

表 3.5. 所需的参数

参数	描述	值
apiVersion	**install-config.yaml** 内容的 API 版本。当前版本是 **v1**。安装程序还可能支持旧的 API 版本。	字符串
baseDomain	云供应商的基域。此基础域用于创建到 OpenShift Container Platform 集群组件的路由。集群的完整 DNS 名称是 **baseDomain** 和 **metadata.name** 参数值的组合，其格式为 **<metadata.name>.<baseDomain>**。	完全限定域名或子域名，如 **example.com**。
metadata	Kubernetes 资源 **ObjectMeta**，其中只消耗 **name** 参数。	对象
metadata.name	集群的名称。集群的 DNS 记录是 **{{.metadata.name}}.{{.baseDomain}}** 的子域。	小写字母,连字符(-)和句点(.)的字符串，如 **dev**。
platform	执行安装的具体平台配置：**aws**、**baremetal**、**azure**、**openstack**、**ovirt**、**vsphere**。有关 **platform.<platform>** 参数的额外信息，请参考下表来了解您的具体平台。	对象

pullSecret	从 Red Hat OpenShift Cluster Manager 获取 pull secret，验证从 Quay.io 等服务中下载 OpenShift Container Platform 组件的容器镜像。	{
		"auths":{
		"cloud.openshift.com":{
		"auth":"b3Blb=",
		"email":"you@example.com"
		},
		"quay.io":{
		"auth":"b3Blb=",
		"email":"you@example.com"
		}
		}
		}

3.5.5.1.2. 网络配置参数

您可以根据现有网络基础架构的要求自定义安装配置。例如，您可以扩展集群网络的 IP 地址块，或者提供不同于默认值的不同 IP 地址块。

只支持 IPv4 地址。

表 3.6. 网络参数

参数	描述	值
networking	集群网络的配置。	对象 **注意** 您不能在安装后修改 **networking** 对象指定的参数。
networking.networkType	要安装的集群网络供应商 Container Network Interface（CNI）插件。	**OpenShiftSDN** 或 **OVNKubernetes**。默认值为 **OpenShiftSDN**。
networking.clusterNetwork	pod 的 IP 地址块。 默认值为 **10.128.0.0/14**，主机前缀为 **/23**。 如果您指定多个 IP 地址块，则块不得互相重叠。	一个对象数组。例如： networking: clusterNetwork: - cidr: 10.128.0.0/14 hostPrefix: 23
networking.clusterNetwork.cidr	使用 **networking.clusterNetwork** 时需要此项。IP 地址块。 一个 IPv4 网络。	使用 CIDR 形式的 IP 地址块。IPv4 块的前缀长度介于 **0** 到 **32** 之间。
networking.clusterNetwork.hostPrefix	分配给每个单独节点的子网前缀长度。例如，如果 **hostPrefix** 设为 **23**，则每个节点从所给的 **cidr** 中分配一个 **/23** 子网。**hostPrefix** 值 **23** 提供 510（2^(32 - 23)- 2）个 pod IP 地址。	子网前缀。 默认值为 **23**。
networking.serviceNetwork	服务的 IP 地址块。默认值为 **172.30.0.0/16**。 OpenShift SDN 和 OVN-Kubernetes 网络供应商只支持服务网络的一个 IP 地址块。	CIDR 格式具有 IP 地址块的数组。例如： networking: serviceNetwork: - 172.30.0.0/16
networking.machineNetwork	机器的 IP 地址块。 如果您指定多个 IP 地址块，则块不得互相重叠。	一个对象数组。例如： networking: machineNetwork: - cidr: 10.0.0.0/16

参数	描述	值
networking.machineNetwork.cidr	使用 **networking.machineNetwork** 时需要。IP 地址块。libvirt 以外的所有平台的默认值为 **10.0.0.0/16**。对于 libvirt，默认值为 **192.168.126.0/24**。	CIDR 表示法中的 IP 网络块。 例如： **10.0.0.0/16**。  **注意** 将 **networking.machineNetwork** 设置为与首选 NIC 所在的 CIDR 匹配。

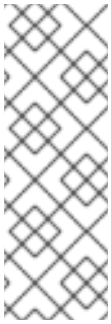
3.5.5.1.3. 可选配置参数

下表描述了可选安装配置参数：

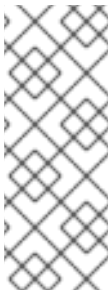
表 3.7. 可选参数

参数	描述	值
additionalTrustBundle	添加到节点可信证书存储中的 PEM 编码 X.509 证书捆绑包。配置了代理时，也可以使用这个信任捆绑包。	字符串
compute	组成计算节点的机器的配置。	machine-pool 对象的数组。详情请查看以下"Machine-pool"表。
compute.architecture	决定池中机器的指令集合架构。目前不支持异构集群，因此所有池都必须指定相同的架构。有效值为 **amd64** （默认值）。	字符串
compute.hyperthreading	是否在计算机器上启用或禁用并发多线程或**超线程**。默认情况下，启用并发多线程以提高机器内核的性能。  **重要** 如果禁用并发多线程，请确保在容量规划时考虑到机器性能可能会显著降低的问题。	**Enabled** 或 **Disabled**
compute.name	使用 **compute** 时需要此值。机器池的名称。	**worker**

参数	描述	值
compute.platform	使用 **compute** 时需要此值。使用此参数指定托管 worker 机器的云供应商。此参数值必须与 **controlPlane.platform** 参数值匹配。	**aws**、**azure**、**gcp**、**openstack**、**ovirt**、**vsphere** 或 **{}**
compute.replicas	要置备的计算机器数量，也称为 worker 机器。	大于或等于 **2** 的正整数。默认值为 **3**。
controlPlane	组成 control plane 的机器的配置。	**MachinePool** 对象的数组。详情请查看以下"Machine-pool"表。
controlPlane.architecture	决定池中机器的指令集合架构。目前不支持异构集群，因此所有池都必须指定相同的架构。有效值为 **amd64** （默认值）。	字符串
controlPlane.hyperthreading	是否在 control plane 机器上启用或禁用并发多线程或**超线程**。默认情况下，启用并发多线程以提高机器内核的性能。  **重要** 如果禁用并发多线程，请确保在容量规划时考虑到机器性能可能会显著降低的问题。	**Enabled** 或 **Disabled**
controlPlane.name	使用 **controlPlane** 时需要。机器池的名称。	**master**
controlPlane.platform	使用 **controlPlane** 时需要。使用此参数指定托管 control plane 机器的云供应商。此参数值必须与 **compute.platform** 参数值匹配。	**aws**、**azure**、**gcp**、**openstack**、**ovirt**、**vsphere** 或 **{}**
controlPlane.replicas	要置备的 control plane 机器数量。	唯一支持的值是 **3**，它是默认值。

参数	描述	值
credentialsMode	Cloud Credential Operator（CCO）模式。如果没有指定任何模式，CCO 会动态地尝试决定提供的凭证的功能，在支持多个模式的平台上使用 mint 模式。  **注意** 不是所有 CCO 模式都支持所有云供应商。如需有关 CCO 模式的更多信息，请参阅*Red Hat Operator 参考指南*内容中的 *Cloud Credential Operator* 条目。	**Mint**、**Passthrough**、**Manual** 或空字符串(**""**)。
fips	启用或禁用 FIPS 模式。默认为 **false**（禁用）。如果启用了 FIPS 模式，运行 OpenShift Container Platform 的 Red Hat Enterprise Linux CoreOS (RHCOS) 机器会绕过默认的 Kubernetes 加密套件，并使用由 RHCOS 提供的加密模块。  **重要** 只有在 **x86_64** 架构中的 OpenShift Container Platform 部署支持 FIPS 验证的/Modules in Process 加密库。  **注意** 如果使用 Azure File 存储，则无法启用 FIPS 模式。	**false** 或 **true**
imageContentSources	release-image 内容的源和仓库。	对象数组。包括一个 **source** 以及可选的 **mirrors**，如下表所示。
imageContentSources.source	使用 **imageContentSources** 时需要。指定用户在镜像拉取规格中引用的仓库。	字符串
imageContentSources.mirrors	指定可能还包含同一镜像的一个或多个仓库。	字符串数组

参数	描述	值
publish	如何发布或公开集群的面向用户的端点，如 Kubernetes API、OpenShift 路由。	**Internal** 或 **External**。把 **publish** 设置为 **Internal** 以部署一个私有集群，它不能被互联网访问。默认值为 **External**。

sshKey	用于验证集群机器访问的 SSH 密钥或密钥。
	 
	注意
	对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 **ssh-agent** 进程使用的 SSH 密钥。
		一个或多个密钥。例如：
		`sshKey:`
		`  <key1>`
		`  <key2>`
		`  <key3>`

3.5.5.1.4. 其他 Azure 配置参数

下表描述了其他 Azure 配置参数：

表 3.8. 其他 Azure 参数

参数	描述	值
compute.platform.azure.osDisk.diskSizeGB	虚拟机的 Azure 磁盘大小。	以 GB 为单位表示磁盘大小的整数。默认值为 **128**。
compute.platform.azure.osDisk.diskType	定义磁盘的类型。	**Standard_LRS**、**Premium_LRS** 或 **标准SSD_LRS**。默认值为 **Premium_LRS**。
controlPlane.platform.azure.osDisk.diskSizeGB	虚拟机的 Azure 磁盘大小。	以 GB 为单位表示磁盘大小的整数。默认值为 **1024**。
controlPlane.platform.azure.osDisk.diskType	定义磁盘的类型。	**Premium_LRS** 或 **标准SSD_LRS**. 默认值为 **Premium_LRS**。
platform.azure.baseDomainResourceGroupName	包含基域的 DNS 区的资源组的名称。	字符串，如 **production_cluster**。

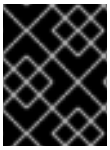
参数	描述	值
platform.azure.outboundType	用于将集群连接到互联网的出站路由策略。如果使用用户定义的路由，则必须在安装集群前配置出站路由。安装程序不负责配置用户定义的路由。	**loadbalancer** 或 **UserDefinedRouting**。默认为 **LoadBalancer**。
platform.azure.region	托管集群的 Azure 区域名称。	任何有效的区域名称，如 **centralus**。
platform.azure.zone	可在其中放入机器的可用区的列表。如需高可用性，请至少指定两个区域。	区域列表，如 **["1", "2", "3"]**。
platform.azure.networkResourceGroupName	包含要将集群部署到的现有 VNet 的资源组名称。这个名称不能和 **platform.azure.baseDomainResourceGroupName**相同。	字符串。
platform.azure.virtualNetwork	要将集群部署到的现有 VNet 的名称。	字符串。
platform.azure.controlPlaneSubnet	要将 control plane 机器部署到的 VNet 中现有子网的名称。	有效的 CIDR，如 **10.0.0.0/16**。
platform.azure.computeSubnet	您要将计算机器部署到的 VNet 中现有子网的名称。	有效的 CIDR，如 **10.0.0.0/16**。
platform.azure.cloudName	用于使用适当 Azure API 端点配置 Azure SDK 的 Azure 云环境名称。如果为空，则使用默认值 **AzurePublicCloud**。	任何有效的云环境，如 **AzurePublicCloud** 或 **AzureUSGovernmentCloud**。

注意

您无法自定义 Azure 可用区，也不能使用标签来整理用于 Azure 集群的 Azure 资源。

3.5.5.2. Azure 的自定义 install-config.yaml 文件示例

您可以自定义 **install-config.yaml** 文件，以指定有关 OpenShift Container Platform 集群平台的更多信息，或修改所需参数的值。



重要

此示例 YAML 文件仅供参考。您必须使用安装程序来获取 **install-config.yaml** 文件，并且修改该文件。

```
apiVersion: v1
baseDomain: example.com (1)
controlPlane: (2)
  hyperthreading: Enabled (3) (4)
  name: master
```

```
    platform:
      azure:
        osDisk:
          diskSizeGB: 1024 5
          diskType: Premium_LRS
        type: Standard_D8s_v3
    replicas: 3
  compute: 6
  - hyperthreading: Enabled 7
    name: worker
    platform:
      azure:
        type: Standard_D2s_v3
        osDisk:
          diskSizeGB: 512 8
          diskType: Standard_LRS
        zones: 9
        - "1"
        - "2"
        - "3"
    replicas: 5
  metadata:
    name: test-cluster 10
  networking: 11
    clusterNetwork:
    - cidr: 10.128.0.0/14
      hostPrefix: 23
    machineNetwork:
    - cidr: 10.0.0.0/16
    networkType: OpenShiftSDN
    serviceNetwork:
    - 172.30.0.0/16
  platform:
    azure:
      baseDomainResourceGroupName: resource_group 12
      region: centralus 13
      resourceGroupName: existing_resource_group 14
      outboundType: Loadbalancer
      cloudName: AzurePublicCloud
  pullSecret: '{"auths": ...}' 15
  fips: false 16
  sshKey: ssh-ed25519 AAAA... 17
```

1 10 13 15 必需。安装程序会提示您输入这个值。

2 6 11 如果没有提供这些参数和值，安装程序会提供默认值。

3 7 **controlPlane** 部分是一个单映射，但 **compute** 部分是一系列映射。为满足不同数据结构的要求，**compute** 部分的第一行必须以连字符 **-** 开头，**controlPlane** 部分的第一行则不可以连字符开头。只使用一个 control plane 池。

4 是否要启用或禁用并发多线程或**超线程**。默认情况下，启用并发多线程以提高机器内核的性能。您可以通过将参数值设为 **Disabled** 来禁用。如果您在某些集群机器上禁用并发多线程，则必须在所有集群机器上禁用。

重要

如果禁用并发多线程，请确保在容量规划时考虑到机器性能可能会显著降低的问题。如果禁用并发多线程，请使用较大的虚拟机类型，如 **Standard_D8s_v3**。

5 8 可以 GB 为单位指定要使用的磁盘大小。control plane 节点（也称为 master 节点）的最低推荐值为 1024 GB。

9 指定要将机器部署到的区域列表。如需高可用性，请至少指定两个区域。

12 指定包含基域的 DNS 区的资源组的名称。

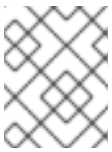
14 指定要安装集群的现有资源组的名称。如果未定义，则会为集群创建新的资源组。

16 是否启用或禁用 FIPS 模式。默认情况下不启用 FIPS 模式。如果启用了 FIPS 模式，运行 OpenShift Container Platform 的 Red Hat Enterprise Linux CoreOS (RHCOS) 机器会绕过默认的 Kubernetes 加密套件，并使用由 RHCOS 提供的加密模块。

重要

只有在 **x86_64** 架构中的 OpenShift Container Platform 部署支持 FIPS 验证的/Modules in Process 加密库。

17 您可以选择提供您用来访问集群中机器的 **sshKey** 值。



注意

对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 **ssh-agent** 进程使用的 SSH 密钥。

3.5.5.3. 在安装过程中配置集群范围代理

生产环境可能会拒绝直接访问互联网，而是提供 HTTP 或 HTTPS 代理。您可以通过在 **install-config.yaml** 文件中配置代理设置，将新的 OpenShift Container Platform 集群配置为使用代理。

先决条件

- 您有一个现有的 **install-config.yaml** 文件。
- 您检查了集群需要访问的站点，并决定是否需要绕过代理。默认情况下代理所有集群出口流量，包括对托管云供应商 API 的调用。您需要将站点添加到 **Proxy** 对象的 **spec.noProxy** 字段来绕过代理。

注意

Proxy 对象 **status.noProxy** 字段使用安装配置中的 **networking.machineNetwork[].cidr**、**networking.clusterNetwork[].cidr** 和 **networking.serviceNetwork[]** 字段的值填充。

对于在 Amazon Web Services(AWS)、Google Cloud Platform(GCP)、Microsoft Azure 和 Red Hat OpenStack Platform(RHOSP)上安装，**Proxy** 对象 **status.noProxy** 字段也会使用实例元数据端点填充(**169.254.169.254**)。

流程

1. 编辑 **install-config.yaml** 文件并添加代理设置。例如：

```
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: example.com 3
additionalTrustBundle: | 4
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
...
```

1. 用于创建集群外 HTTP 连接的代理 URL。URL 必须是 **http**。
2. 用于创建集群外 HTTPS 连接的代理 URL。
3. 要排除在代理中的目标域名、IP 地址或其他网络 CIDR 的逗号分隔列表。在域前面加 **.** 来仅匹配子域。例如： **.y.com** 匹配 **x.y.com**，但不匹配 **y.com**。使用 * 绕过所有目的地的代理。
4. 如果提供，安装程序会在 **openshift-config 命名空间中生成名为 user-ca- bundle** 的配置映射来保存额外的 CA 证书。如果您提供 **additionalTrustBundle** 和至少一个代理设置，则 **Proxy** 对象会被配置为引用 **trustedCA** 字段中的 **user-ca-bundle** 配置映射。然后，Cluster Network Operator 会创建一个 **trusted-ca-bundle** 配置映射，该配置映射将为 **trustedCA** 参数指定的内容与 RHCOS 信任捆绑包合并。 **additionalTrustBundle** 字段是必需的，除非代理的身份证书由来自 RHCOS 信任捆绑包的颁发机构签名。

注意

安装程序不支持代理的 **readinessEndpoints** 字段。

2. 保存该文件，并在安装 OpenShift Container Platform 时引用。

安装程序会创建一个名为 **cluster** 的集群范围代理，该代理使用提供的 **install-config.yaml** 文件中的代理设置。如果没有提供代理设置，仍然会创建一个 **cluster Proxy** 对象，但它会有一个空 **spec**。

注意

只支持名为 **cluster** 的 **Proxy** 对象，且无法创建额外的代理。

3.5.6. 网络配置阶段

当在安装前指定集群配置时，在安装过程中的几个阶段可以修改网络配置：

阶段 1

输入 **openshift-install create install-config** 命令后。在 **install-config.yaml** 文件中，您可以自定义以下与网络相关的字段：

- **networking.networkType**

- **networking.clusterNetwork**
- **networking.serviceNetwork**
- **networking.machineNetwork**
  有关这些字段的更多信息，请参阅"安装配置参数"。

注意

将 **networking.machineNetwork** 设置为与首选 NIC 所在的 CIDR 匹配。

阶段 2

输入 **openshift-install create manifests** 命令后。如果必须指定高级网络配置，在这个阶段中，只能使用您要修改的字段来定义自定义的 Cluster Network Operator 清单。

在 2 阶段，您无法覆盖 **install-config.yaml** 文件中的 1 阶段中指定的值。但是，您可以在第 2 阶段进一步自定义集群网络供应商。

3.5.7. 指定高级网络配置

您可以通过为集群网络供应商指定额外的配置，使用高级配置自定义将集群整合到现有网络环境中。您只能在安装集群前指定高级网络配置。

重要

不支持修改安装程序创建的 OpenShift Container Platform 清单文件。支持应用您创建的清单文件，如以下流程所示。

先决条件

- 创建 **install-config.yaml** 文件并完成对其所做的任何修改。

流程

1. 进入包含安装程序的目录并创建清单：

```
$ ./openshift-install create manifests --dir <installation_directory>
```

其中：

<installation_directory>

指定包含集群的 **install-config.yaml** 文件的目录名称。

2. 在 **<installation_directory>/manifests/** 目录下，为高级网络配置创建一个名为 **cluster-network-03-config.yml** 的 stub 清单文件：

```
$ cat <<EOF > <installation_directory>/manifests/cluster-network-03-config.yml
apiVersion: operator.openshift.io/v1
kind: Network
metadata:
  name: cluster
spec:
EOF
```

其中：

<installation_directory>
指定包含集群的 **manifests/** 目录的目录名称。

3. 在编辑器中打开 **cluster-network-03-config.yml** 文件，并为集群指定高级网络配置，如下例所示：

为 OpenShift SDN 网络供应商指定不同的 VXLAN 端口

```
apiVersion: operator.openshift.io/v1
kind: Network
metadata:
  name: cluster
spec:
  defaultNetwork:
    openshiftSDNConfig:
      vxlanPort: 4800
```

4. 保存 **cluster-network-03-config.yml** 文件，再退出文本编辑器。

5. 可选：备份 **manifests/cluster-network-03-config.yml** 文件。创建集群时，安装程序会删除 **manifests/** 目录。

3.5.8. Cluster Network Operator 配置

集群网络的配置作为 Cluster Network Operator (CNO) 配置的一部分被指定，并存储在名为 **cluster**的自定义资源（CR）对象中。CR 指定 **operator.openshift.io** API 组中的 **Network** API 的字段。

CNO 配置会在集群安装过程中从 **Network.config.openshift.io** API 组中的 **Network** API 继承以下字段，这些字段无法更改：

clusterNetwork
从中分配 pod IP 地址的 IP 地址池。

serviceNetwork
服务的 IP 地址池。

defaultNetwork.type
集群网络供应商，如 OpenShift SDN 或 OVN-Kubernetes。

您可以通过在名为 **cluster** 的 CNO 对象中设置 **defaultNetwork** 对象的字段来为集群指定集群网络供应商配置。

3.5.8.1. Cluster Network Operator 配置对象

Cluster Network Operator（CNO）的字段在下表中描述：

表 3.9. Cluster Network Operator 配置对象

字段	类型	描述
metadata.name	**字符串**	CNO 对象的名称。这个名称始终是 **cluster**。

字段	类型	描述
spec.clusterNetwork	**数组**	用于指定从哪些 IP 地址块分配 Pod IP 地址以及分配给集群中每个节点的子网前缀长度的列表。例如： spec: clusterNetwork: - cidr: 10.128.0.0/19 hostPrefix: 23 - cidr: 10.128.32.0/19 hostPrefix: 23 此值是只读的，并在 **install-config.yaml** 文件中指定。
spec.serviceNetwork	**数组**	服务的 IP 地址块。OpenShift SDN 和 OVN-Kubernetes Container Network Interface（CNI）网络供应商只支持服务网络具有单个 IP 地址块。例如： spec: serviceNetwork: - 172.30.0.0/14 此值是只读的，并在 **install-config.yaml** 文件中指定。
spec.defaultNetwork	**对象**	为集群网络配置 Container Network Interface（CNI）集群网络供应商。
spec.kubeProxyConfig	**对象**	此对象的字段指定 kube-proxy 配置。如果您使用 OVN-Kubernetes 集群网络供应商，则 kube-proxy 的配置不会起作用。

defaultNetwork 对象配置

defaultNetwork 对象的值在下表中定义：

表 3.10. defaultNetwork 对象

字段	类型	描述
type	**字符串**	**OpenShiftSDN** 或 **OVNKubernetes**。在安装过程中选择了集群网络供应商。集群安装后无法更改这个值。 **注意** OpenShift Container Platform 默认使用 OpenShift SDN Container Network Interface（CNI）集群网络供应商。

字段	类型	描述
openshiftSDNConfig	**对象**	此对象仅对 OpenShift SDN 集群网络供应商有效。
ovnKubernetesConfig	**对象**	此对象仅对 OVN-Kubernetes 集群网络供应商有效。

配置 OpenShift SDN CNI 集群网络供应商
下表描述了 OpenShift SDN Container Network Interface（CNI）集群网络供应商的配置字段。

表 3.11. openshiftSDNConfig 对象

字段	类型	描述
mode	**字符串**	配置 OpenShift SDN 的网络隔离模式。默认值为 **NetworkPolicy**。 **Multitenant** 和 **Subnet** 的值可以向后兼容 OpenShift Container Platform 3.x，但不推荐这样做。集群安装后无法更改这个值。
mtu	**整数**	VXLAN 覆盖网络的最大传输单元 (MTU) 。这根据主网络接口的 MTU 自动探测。您通常不需要覆盖检测到的 MTU。 如果自动探测的值不是您期望的，请确认节点上主网络接口中的 MTU 是正确的。您不能使用这个选项更改节点上主网络接口的 MTU 值。 如果您的集群中的不同节点需要不同的 MTU 值，则必须将此值设置为比集群中的最低 MTU 值小 **50**。例如，如果集群中的某些节点的 MTU 为 **9001**，而某些节点的 MTU 为 **1500**，则必须将此值设置为 **1450**。 集群安装后无法更改这个值。
vxlanPort	**整数**	用于所有 VXLAN 数据包的端口。默认值为 **4789**。集群安装后无法更改这个值。 如果您在虚拟环境中运行，并且现有节点是另一个 VXLAN 网络的一部分，那么可能需要更改此值。例如，当在 VMware NSX-T 上运行 OpenShift SDN 覆盖时，您必须为 VXLAN 选择一个备用端口，因为两个 SDN 都使用相同的默认 VXLAN 端口号。 在 Amazon Web Services (AWS) 上，您可以在端口 **9000** 和端口 **9999** 之间为 VXLAN 选择一个备用端口。

OpenShift SDN 配置示例

```
defaultNetwork:
  type: OpenShiftSDN
  openshiftSDNConfig:
```

```
    mode: NetworkPolicy
    mtu: 1450
    vxlanPort: 4789
```

配置 OVN-Kubernetes CNI 集群网络供应商
下表描述了 OVN-Kubernetes CNI 集群网络供应商的配置字段。

表 3.12. ovnKubernetesConfig 对象

字段	类型	描述
mtu	**整数**	Geneve（Generic Network Virtualization Encapsulation）覆盖网络的最大传输单元（MTU）。这根据主网络接口的 MTU 自动探测。您通常不需要覆盖检测到的 MTU。 如果自动探测的值不是您期望的，请确认节点上主网络接口中的 MTU 是正确的。您不能使用这个选项更改节点上主网络接口的 MTU 值。 如果您的集群中的不同节点需要不同的 MTU 值，则必须将此值设置为比集群中的最低 MTU 值小 **100**。例如，如果集群中的某些节点的 MTU 为 **9001**，而某些节点的 MTU 为 **1500**，则必须将此值设置为 **1400**。 集群安装后无法更改这个值。
genevePort	**整数**	用于所有 Geneve 数据包的端口。默认值为 **6081**。集群安装后无法更改这个值。

OVN-Kubernetes 配置示例

```
defaultNetwork:
  type: OVNKubernetes
  ovnKubernetesConfig:
    mtu: 1400
    genevePort: 6081
```

kubeProxyConfig 对象配置
kubeProxyConfig 对象的值在下表中定义：

表 3.13. kubeProxyConfig 对象

字段	类型	描述

字段	类型	描述
iptablesSyncPeriod	**字符串**	**iptables** 规则的刷新周期。默认值为 **30s**。有效的后缀包括 **s**、**m** 和 **h**，具体参见 Go **time** 软件包文档。 **注意** 由于 OpenShift Container Platform 4.3 及更高版本中引进了性能上的改进，现在不再需要调整 **iptablesSyncPeriod** 参数。

proxyArguments.iptables-min-sync-period	**数组**	刷新 **iptables** 规则前的最短时长。此字段确保刷新的频率不会过于频繁。有效的后缀包括 **s**、**m** 和 **h**，具体参见 Go **time** 软件包。默认值为：
		
		`kubeProxyConfig:`
		`  proxyArguments:`
		`  iptables-min-sync-period:`
		`  - 0s`

3.5.9. 使用 OVN-Kubernetes 配置混合网络

您可以将集群配置为使用 OVN-Kubernetes 的混合网络。这允许支持不同节点网络配置的混合集群。例如：集群中运行 Linux 和 Windows 节点时需要这样做。

重要

您必须在安装集群过程中使用 OVN-Kubernetes 配置混合网络。您不能在安装过程中切换到混合网络。

先决条件

- 您在 **install-config.yaml** 文件中为 **networking.networkType** 参数定义了 **OVNKubernetes**。如需更多信息，请参阅有关在所选云供应商上配置 OpenShift Container Platform 网络自定义的安装文档。

流程

1. 进入包含安装程序的目录并创建清单：

```
$ ./openshift-install create manifests --dir <installation_directory>
```

其中：

<installation_directory>
指定包含集群的 **install-config.yaml** 文件的目录名称。

2. 在 **<installation_directory>/manifests/** 目录下，为高级网络配置创建一个名为 **cluster-network-03-config.yml** 的 stub 清单文件：

```
$ cat <<EOF > <installation_directory>/manifests/cluster-network-03-config.yml
apiVersion: operator.openshift.io/v1
kind: Network
metadata:
  name: cluster
spec:
EOF
```

其中：

<installation_directory>

指定包含集群的 **manifests/** 目录的目录名称。

3. 在编辑器中打开 **cluster-network-03-config.yml** 文件，并使用混合网络配置 OVN-Kubernetes，如下例所示：

指定混合网络配置

```
apiVersion: operator.openshift.io/v1
kind: Network
metadata:
  name: cluster
spec:
  defaultNetwork:
    ovnKubernetesConfig:
      hybridOverlayConfig:
        hybridClusterNetwork: 1
        - cidr: 10.132.0.0/14
          hostPrefix: 23
        hybridOverlayVXLANPort: 9898 2
```

1 指定用于额外覆盖网络上节点的 CIDR 配置。**hybridClusterNetwork** CIDR 无法与 **clusterNetwork** CIDR 重叠。

2 为额外覆盖网络指定自定义 VXLAN 端口。这是在 vSphere 上安装的集群中运行 Windows 节点所需要的，且不得为任何其他云供应商配置。自定义端口可以是除默认 **4789** 端口外的任何打开的端口。有关此要求的更多信息，请参阅 Microsoft 文档中的 Pod 到主机间的 pod 连接性。

4. 保存 **cluster-network-03-config.yml** 文件，再退出文本编辑器。

5. 可选：备份 **manifests/cluster-network-03-config.yml** 文件。创建集群时，安装程序会删除 **manifests/** 目录。

注意

有关在同一集群中使用 Linux 和 Windows 节点的更多信息，请参阅了解 Windows 容器工作负载。

3.5.10. 部署集群

您可以在兼容云平台中安装 OpenShift Container Platform。

重要

安装程序的 **create cluster** 命令只能在初始安装过程中运行一次。

先决条件

- 配置托管集群的云平台的帐户。
- 获取 OpenShift Container Platform 安装程序以及集群的 pull secret。

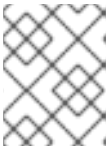
流程

1. 更改为包含安装程序的目录并初始化集群部署：

```
$ ./openshift-install create cluster --dir <installation_directory> \ 1
    --log-level=info 2
```

1 对于 **<installation_directory>**，请指定自定义 **./install-config.yaml** 文件的位置。

2 要查看不同的安装详情，请指定 **warn**、**debug** 或 **error**，而不要指定 **info**。



注意

如果您在主机上配置的云供应商帐户没有足够的权限来部署集群，安装过程将会停止，并且显示缺少的权限。

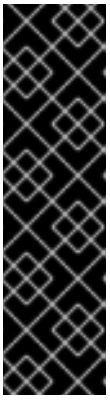
集群部署完成后，终端会显示访问集群的信息，包括指向其 Web 控制台的链接和 **kubeadmin** 用户的凭证。

输出示例

```
...
INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run 'export
KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'
INFO Access the OpenShift web-console here: https://console-openshift-
console.apps.mycluster.example.com
INFO Login to the console with user: "kubeadmin", and password: "4vYBz-Ee6gm-ymBZj-
Wt5AL"
INFO Time elapsed: 36m22s
```

注意

当安装成功时，集群访问和凭证信息还会输出到 **<installation_directory>/.openshift_install.log**。



重要

- 安装程序生成的 Ignition 配置文件包含在 24 小时后过期的证书，然后在过期时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外情况是，您需要手动批准待处理的 **node-bootstrapper** 证书签名请求（CSR）来恢复 kubelet 证书。如需更多信息，请参阅*从过期的 control plane 证书中恢复*的文档。
- 建议您在生成 12 小时后使用 Ignition 配置文件，因为集群安装后 24 小时证书从 16 小时轮转至 22 小时。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中运行证书更新时避免安装失败。

重要

您不得删除安装程序或安装程序所创建的文件。需要这两者才能删除集群。

3.5.11. 通过下载二进制文件安装 OpenShift CLI

您需要安装 CLI（**oc**） 来使用命令行界面与 OpenShift Container Platform 进行交互。您可在 Linux 、Windows 或 macOS 上安装 **oc**。

重要

如果安装了旧版本的 **oc**，则无法使用 OpenShift Container Platform 4.6 中的所有命令。下载并安装新版本的 **oc**。

3.5.11.1. 在 Linux 上安装 OpenShift CLI

您可以按照以下流程在 Linux 上安装 OpenShift CLI（**oc**）二进制文件。

流程

1. 进入到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 在 **Version** 下拉菜单中选择相应的版本。
3. 单击 **OpenShift v4.6 Linux 客户端** 条目旁边的 **Download Now**，再保存文件。
4. 解包存档：

```
$ tar xvzf <file>
```

5. 把 **oc** 二进制代码放到 **PATH** 中的目录中。
   执行以下命令可以查看当前的 **PATH** 设置：

```
$ echo $PATH
```

安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
$ oc <command>
```

3.5.11.2. 在 Windows 上安装 OpenShift CLI

您可以按照以下流程在 Windows 上安装 OpenShift CLI（**oc**）二进制代码。

流程

1. 进入到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 在 **Version** 下拉菜单中选择相应的版本。
3. 单击 **OpenShift v4.6 Windows 客户端**条目旁边的 **Download Now**，再保存文件。
4. 使用 ZIP 程序解压存档。
5. 把 **oc** 二进制代码放到 **PATH** 中的目录中。
   要查看您的 **PATH**，请打开命令提示窗口并执行以下命令：

```
C:\> path
```

安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
C:\> oc <command>
```

3.5.11.3. 在 macOS 上安装 OpenShift CLI

您可以按照以下流程在 macOS 上安装 OpenShift CLI（**oc**）二进制代码。

流程

1. 进入到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 在 **Version** 下拉菜单中选择相应的版本。
3. 单击 **OpenShift v4.6 MacOSX 客户端**条目旁边的 **Download Now**，再保存文件。
4. 解包和解压存档。
5. 将 **oc** 二进制文件移到 PATH 的目录中。
   要查看您的 **PATH**，打开一个终端窗口并执行以下命令：

```
$ echo $PATH
```

安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
$ oc <command>
```

3.5.12. 使用 CLI 登录到集群

您可以通过导出集群 **kubeconfig** 文件，以默认系统用户身份登录集群。**kubeconfig** 文件包含关于集群的信息，供 CLI 用于将客户端连接到正确集群和 API 服务器。该文件特只适用于一个特定的集群，在 OpenShift Container Platform 安装过程中创建。

先决条件

- 已部署了 OpenShift Container Platform 集群。

- 已安装 **oc** CLI。

流程

1. 导出 **kubeadmin** 凭证：

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

2. 使用导出的配置，验证能否成功运行 **oc** 命令：

```
$ oc whoami
```

输出示例

```
system:admin
```

其他资源

- 如需有关访问和了解 OpenShift Container Platform Web 控制台的更多信息，请参阅访问 Web 控制台。

3.5.13. OpenShift Container Platform 的 Telemetry 访问

在 OpenShift Container Platform 4.6 中，默认运行的 Telemetry 服务提供有关集群健康状况和成功更新的指标，需要访问互联网。如果您的集群连接到互联网，Telemetry 会自动运行，而且集群会注册到 OpenShift Cluster Manager。

确认 OpenShift Cluster Manager 清单正确后，可以由 Telemetry 自动维护，也可以使用 OpenShift Cluster Manager 手动维护，使用订阅监控来跟踪帐户或多集群级别的 OpenShift Container Platform 订阅。

其他资源

- 有关 Telemetry 服务的更多信息，请参阅关于远程健康监控。

3.5.14. 后续步骤

- 自定义集群。
- 若有需要，您可以选择不使用远程健康报告。

3.6. 将 AZURE 上的集群安装到现有的 VNET

在 OpenShift Container Platform 版本 4.6 中，您可以在 Microsoft Azure 上将集群安装到现有 Azure Virtual Network（VNet）中。安装程序会置备所需基础架构的其余部分，您可以进一步定制这些基础架构。要自定义安装，请在安装集群前修改 **install-config.yaml** 文件中的参数。

3.6.1. 先决条件

- 查看有关 OpenShift Container Platform 安装和更新流程的详细信息。

- 配置一个 Azure 帐户以托管集群，并决定要将集群部署到的已测试和验证的区域。
- 如果使用防火墙，则必须将其配置为允许集群需要访问的站点。
- 如果不允许系统管理身份和访问管理（IAM），集群管理员可以 手动创建和维护 IAM 凭证。手动模式也可以用于云 IAM API 无法访问的环境中。

3.6.2. 关于为 OpenShift Container Platform 集群重复使用 VNet

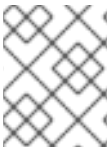
在 OpenShift Container Platform 4.6 中，您可以在 Microsoft Azure 中将集群部署到现有的 Azure Virtual Network（VNet）中。如果您这样做，还必须在 VNet 和路由规则中使用现有子网。

通过将 OpenShift Container Platform 部署到现有的 Azure VNet 中，您可能会避开新帐户中的服务限制，或者更容易地利用公司所设置的操作限制。如果您无法获得创建 VNet 所需的基础架构创建权限，则可以使用这个选项。

3.6.2.1. 使用 VNet 的要求

当使用现有 VNet 部署集群时，必须在安装集群前执行额外网络配置。在安装程序置备的基础架构集群中，安装程序通常会创建以下组件，但在安装到现有 VNet 时不会创建它们：

- 子网
- 路由表
- VNets
- 网络安全组



注意

安装程序要求您使用由云提供的 DNS 服务器。不支持使用自定义 DNS 服务器，并导致安装失败。

如果您使用自定义 VNet，您必须正确配置它及其子网，以便安装程序和集群使用。安装程序不能为集群分配要使用的网络范围，为子网设置路由表，或者设置类似 DHCP 的 VNet 选项，因此您必须在安装集群前配置它们。

集群必须能够访问包含现有 VNet 和子网的资源组。虽然集群创建的所有资源都放在它创建的单独资源组中，但有些网络资源则从另外一个独立的组中使用。一些集群 Operator 必须能够访问这两个资源组中的资源。例如，Machine API 控制器会为它创建的虚拟机附加 NICS，使其从网络资源组中划分子网。

您的 VNet 必须满足以下特征：

- VNet 的 CIDR 块必须包含 **Networking.machineCIDR**，它是集群机器的 IP 地址池。
- VNet 及其子网必须属于同一资源组，子网必须配置为使用 Azure 分配的 DHCP IP 地址而不是静态 IP 地址。

您必须在 VNet 中提供两个子网，一个用于 control plane 机器，一个用于计算机器。因为 Azure 在您指定的区域内的不同可用区中分发机器，所以集群将默认具有高可用性功能。

要确保您提供的子网适合您的环境，安装程序会确认以下信息：

- 所有指定的子网都存在。

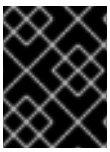
- 有两个专用子网，一个用于 control plane 机器，一个用于计算机器。
- 子网 CIDR 属于您指定的机器 CIDR。机器不会在没有为其提供私有子网的可用区中置备。如果需要，安装程序会创建管理 control plane 和 worker 节点的公共负载均衡器，Azure 会为其分配一个公共 IP 地址。

注意

如果您销毁了使用现有 VNet 的集群，则不会删除 VNet。

3.6.2.1.1. 网络安全组要求

托管 compute 和 control plane 机器的子网的网络安全组需要特定的访问权限，以确保集群通信正确。您必须创建规则来允许访问所需的集群通信端口。

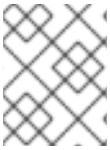


重要

在安装集群前必须先设置网络安全组规则。如果您试图在没有所需访问权限的情况下安装集群，安装程序就无法访问 Azure API，且会导致安装失败。

表 3.14. 所需端口

端口	描述	Control plane	Compute
80	允许 HTTP 流量		x
443	允许 HTTPS 流量		x
6443	允许与 control plane 机器通信。	x	
22623	允许与机器配置服务器通信。	x	



注意

由于集群组件不会修改用户提供的网络安全组（Kubernetes 控制器更新它），因此会创建一个伪网络安全组，供 Kubernetes 控制器修改，而不影响其余的环境。

3.6.2.2. 权限划分

从 OpenShift Container Platform 4.3 开始，您不需要安装程序置备的基础架构集群部署所需的所有权限。这与您所在机构可能已有的权限划分类似：不同的个人可以在您的云中创建不同的资源。例如，您可以创建针对于特定应用程序的对象，如实例、存储和负载均衡器，但不能创建与网络相关的组件，如 VNets 、子网或入站规则。

您在创建集群时使用的 Azure 凭证不需要 VNets 和核心网络组件（如子网、路由表、互联网网关、NAT 和 VPN）所需的网络权限。您仍然需要获取集群中的机器需要的应用程序资源的权限，如负载均衡器 、安全组、存储帐户和节点。

3.6.2.3. 集群间隔离

因为集群无法修改现有子网中的网络安全组，所以无法在 VNet 中相互隔离集群。

3.6.3. OpenShift Container Platform 的互联网访问

在 OpenShift Container Platform 4.6 中，您需要访问互联网来安装集群。

您必须具有以下互联网访问权限：

- 访问 OpenShift Cluster Manager 以下载安装程序并执行订阅管理。如果集群可以访问互联网，并且没有禁用 Telemetry，该服务会自动授权您的集群。
- 访问 Quay.io，以获取安装集群所需的软件包。
- 获取执行集群更新所需的软件包。

重要

如果您的集群无法直接访问互联网，则可以在置备的某些类基础架构上执行受限网络安装。在此过程中，您要下载所需的内容，并使用它在镜像 registry（mirror registry） 中填充安装集群并生成安装程序所需的软件包。对于某些安装类型，集群要安装到的环境不需要访问互联网。在更新集群之前，要更新 registry 镜像系统中的内容。

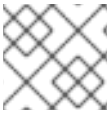
3.6.4. 生成 SSH 私钥并将其添加到代理中

如果要在集群上执行安装调试或灾难恢复，则必须为 **ssh-agent** 和安装程序提供 SSH 密钥。您可以使用此密钥访问公共集群中的 bootstrap 机器来排除安装问题。

注意

在生产环境中，您需要进行灾难恢复和调试。

您可以使用此密钥以 **core** 用户身份通过 SSH 连接到 master 节点。在部署集群时，此密钥会添加到 **core** 用户的 **~/.ssh/authorized_keys** 列表中。



注意

您必须使用一个本地密钥，而不要使用在特定平台上配置的密钥，如 AWS 密钥对。

流程

1. 如果还没有为计算机上免密码身份验证而配置的 SSH 密钥，请创建一个。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ ssh-keygen -t ed25519 -N '' \
  -f <path>/<file_name> 1
```

1 指定新 SSH 密钥的路径和文件名，如 **~/.ssh/id_rsa**。如果您已有密钥对，请确保您的公钥位于 **~/.ssh** 目录中。

运行此命令会在指定的位置生成不需要密码的 SSH 密钥。

注意

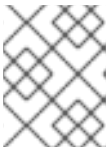
如果您计划在 **x86_64** 架构中安装使用 FIPS 验证的/Modules in Process 加密库的 OpenShift Container Platform 集群，不要创建使用 **ed25519** 算法的密钥。反之，创建一个使用 **rsa** 或 **ecdsa** 算法的密钥。

2. 作为后台任务启动 **ssh-agent** 进程：

```
$ eval "$(ssh-agent -s)"
```

输出示例

```
Agent pid 31874
```



注意

如果您的集群采用 FIPS 模式，则只使用 FIPS 兼容算法来生成 SSH 密钥。密钥必须是 RSA 或 ECDSA。

3. 将 SSH 私钥添加到 **ssh-agent**：

```
$ ssh-add <path>/<file_name> 1
```

输出示例

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

1 指定 SSH 私钥的路径和文件名，如 **~/.ssh/id_rsa**

后续步骤

- 在安装 OpenShift Container Platform 时，为安装程序提供 SSH 公钥。

3.6.5. 获取安装程序

在安装 OpenShift Container Platform 之前，将安装文件下载到本地计算机上。

先决条件

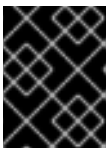
- 运行 Linux 或 macOS 的计算机，本地磁盘空间为 500 MB

流程

1. 访问 OpenShift Cluster Manager 站点的 Infrastructure Provider 页面。如果您有红帽帐号，请使用自己的凭证登录。如果没有，请创建一个帐户。
2. 选择您的基础架构供应商。
3. 进入适用于您的安装类型的页面，下载您的操作系统的安装程序，并将文件放在要保存安装配置文件的目录中。。

重要

安装程序会在用来安装集群的计算机上创建若干文件。在完成集群安装后，您必须保留安装程序和安装程序所创建的文件。这两个文件都需要删除集群。



重要

删除安装程序创建的文件不会删除您的集群，即使集群在安装过程中失败也是如此。要删除集群，为特定云供应商完成 OpenShift Container Platform 卸载流程。

4. 提取安装程序。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ tar xvf openshift-install-linux.tar.gz
```

5. 从 Red Hat OpenShift Cluster Manager 下载安装 pull secret 。通过此 pull secret，您可以进行所含授权机构提供的服务的身份验证，这些服务包括为 OpenShift Container Platform 组件提供容器镜像的 Quay.io。

3.6.6. 创建安装配置文件

您可以自定义在 Microsoft Azure 上安装的 OpenShift Container Platform 集群。

先决条件

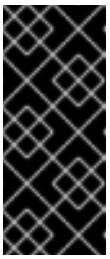
- 获取 OpenShift Container Platform 安装程序以及集群的 pull secret。

流程

1. 创建 **install-config.yaml** 文件。
   a. 更改到包含安装程序的目录，再运行以下命令：

```
$ ./openshift-install create install-config --dir <installation_directory> 1
```

   1 对于 **<installation_directory>**，请指定用于保存安装程序所创建的文件的目录名称。



重要

指定一个空目录。一些安装信息，如 bootstrap X.509 证书，有较短的过期间隔，因此不要重复使用安装目录。如果要重复使用另一个集群安装中的个别文件，可以将其复制到您的目录中。但是，一些安装数据的文件名可能会在发行版本之间有所改变。从 OpenShift Container Platform 老版本中复制安装文件时要格外小心。

   b. 在提示符处，提供您的云的配置详情：
      i. 可选：选择用来访问集群机器的 SSH 密钥。

注意

对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 **ssh-agent** 进程使用的 SSH 密钥。

ii. 选择 **azure** 作为目标平台。

iii. 如果计算机上没有 Microsoft Azure 配置集，请为您的订阅和服务主体指定以下 Azure 参数值：

- **azure subscription id**：要用于集群的订阅 ID。指定帐户输出中的 **id** 值。
- **azure tenant id**：租户 ID。指定帐户输出中的 **tenantId** 值。
- **azure service principal client id**：服务主体的 **appId** 参数值。
- **azure service principal client secret**：服务主体的 **password** 参数值。

iv. 选择要在其中部署集群的区域。

v. 选择集群要部署到的基域。基域与您为集群创建的 Azure DNS 区对应。

vi. 为集群输入一个描述性名称。

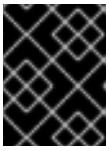
重要

所有通过公共端点提供的 Azure 资源均存在资源名称的限制，您无法创建使用某些名称的资源。如需 Azure 限制词语列表，请参阅 Azure 文档中的解决保留资源名称错误。

vii. 粘贴 Red Hat OpenShift Cluster Manager 中的 pull secret 。

2. 修改 **install-config.yaml** 文件。您可以在**安装配置参数**部分中找到有关可用参数的更多信息。

3. 备份 **install-config.yaml** 文件，以便用于安装多个集群。

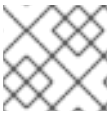


重要

install-config.yaml 文件会在安装过程中消耗掉。如果要重复使用此文件，必须现在备份。

3.6.6.1. 安装配置参数

在部署 OpenShift Container Platform 集群前，您可以提供参数值，以描述托管集群的云平台的帐户并选择性地自定义集群平台。在创建 **install-config.yaml** 安装配置文件时，您可以通过命令行来提供所需的参数的值。如果要自定义集群，可以修改 **install-config.yaml** 文件来提供关于平台的更多信息。



注意

安装之后，您无法修改 **install-config.yaml** 文件中的这些参数。

重要

openshift-install 命令不验证参数的字段名称。如果指定了不正确的名称，则不会创建相关的文件或对象，且不会报告错误。确保所有指定的参数的字段名称都正确。

3.6.6.1.1. 所需的配置参数

下表描述了所需的安装配置参数：

表 3.15. 所需的参数

参数	描述	值
apiVersion	**install-config.yaml** 内容的 API 版本。当前版本是 **v1**。安装程序还可能支持旧的 API 版本。	字符串
baseDomain	云供应商的基域。此基础域用于创建到 OpenShift Container Platform 集群组件的路由。集群的完整 DNS 名称是 **baseDomain** 和 **metadata.name** 参数值的组合，其格式为 **<metadata.name>.<baseDomain>**。	完全限定域名或子域名，如 **example.com**。
metadata	Kubernetes 资源 **ObjectMeta**，其中只消耗 **name** 参数。	对象
metadata.name	集群的名称。集群的 DNS 记录是 **{{.metadata.name}}.{{.baseDomain}}** 的子域。	小写字母,连字符(-)和句点(.)的字符串，如 **dev**。
platform	执行安装的具体平台配置：**aws**、**baremetal**、**azure**、**openstack**、**ovirt**、**vsphere**。有关 **platform.<platform>** 参数的额外信息，请参考下表来了解您的具体平台。	对象

pullSecret	从 Red Hat OpenShift Cluster Manager 获取 pull secret，验证从 Quay.io 等服务中下载 OpenShift Container Platform 组件的容器镜像。	{
		"auths":{
		"cloud.openshift.com":{
		"auth":"b3Blb=",
		"email":"you@example.com"
		},
		"quay.io":{
		"auth":"b3Blb=",
		"email":"you@example.com"
		}
		}
		}

3.6.6.1.2. 网络配置参数

您可以根据现有网络基础架构的要求自定义安装配置。例如，您可以扩展集群网络的 IP 地址块，或者提供不同于默认值的不同 IP 地址块。

只支持 IPv4 地址。

表 3.16. 网络参数

参数	描述	值
networking	集群网络的配置。	对象 **注意** 您不能在安装后修改 **networking** 对象指定的参数。
networking.networkType	要安装的集群网络供应商 Container Network Interface（CNI）插件。	**OpenShiftSDN** 或 **OVNKubernetes**。默认值为 **OpenShiftSDN**。
networking.clusterNetwork	pod 的 IP 地址块。 默认值为 **10.128.0.0/14**，主机前缀为 **/23**。 如果您指定多个 IP 地址块，则块不得互相重叠。	一个对象数组。例如： networking: clusterNetwork: - cidr: 10.128.0.0/14 hostPrefix: 23
networking.clusterNetwork.cidr	使用 **networking.clusterNetwork** 时需要此项。IP 地址块。 一个 IPv4 网络。	使用 CIDR 形式的 IP 地址块。IPv4 块的前缀长度介于 **0** 到 **32** 之间。
networking.clusterNetwork.hostPrefix	分配给每个单独节点的子网前缀长度。例如，如果 **hostPrefix** 设为 **23**，则每个节点从所给的 **cidr** 中分配一个 **/23** 子网。**hostPrefix** 值 **23** 提供 510（2^(32 - 23)- 2）个 pod IP 地址。	子网前缀。 默认值为 **23**。
networking.serviceNetwork	服务的 IP 地址块。默认值为 **172.30.0.0/16**。 OpenShift SDN 和 OVN-Kubernetes 网络供应商只支持服务网络的一个 IP 地址块。	CIDR 格式具有 IP 地址块的数组。例如： networking: serviceNetwork: - 172.30.0.0/16
networking.machineNetwork	机器的 IP 地址块。 如果您指定多个 IP 地址块，则块不得互相重叠。	一个对象数组。例如： networking: machineNetwork: - cidr: 10.0.0.0/16

参数	描述	值
networking.machineNetwork.cidr	使用 **networking.machineNetwork** 时需要。IP 地址块。libvirt 以外的所有平台的默认值为 **10.0.0.0/16**。对于 libvirt，默认值为 **192.168.126.0/24**。	CIDR 表示法中的 IP 网络块。 例如： **10.0.0.0/16**。  **注意** 将 **networking.machineNetwork** 设置为与首选 NIC 所在的 CIDR 匹配。

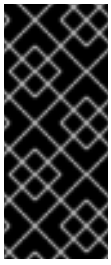
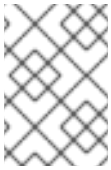
3.6.6.1.3. 可选配置参数

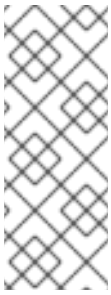
下表描述了可选安装配置参数：

表 3.17. 可选参数

参数	描述	值
additionalTrustBundle	添加到节点可信证书存储中的 PEM 编码 X.509 证书捆绑包。配置了代理时，也可以使用这个信任捆绑包。	字符串
compute	组成计算节点的机器的配置。	machine-pool 对象的数组。详情请查看以下"Machine-pool"表。
compute.architecture	决定池中机器的指令集合架构。目前不支持异构集群，因此所有池都必须指定相同的架构。有效值为 **amd64** （默认值）。	字符串
compute.hyperthreading	是否在计算机器上启用或禁用并发多线程或**超线程**。默认情况下，启用并发多线程以提高机器内核的性能。  **重要** 如果禁用并发多线程，请确保在容量规划时考虑到机器性能可能会显著降低的问题。	**Enabled** 或 **Disabled**
compute.name	使用 **compute** 时需要此值。机器池的名称。	**worker**

参数	描述	值
compute.platform	使用 **compute** 时需要此值。使用此参数指定托管 worker 机器的云供应商。此参数值必须与 **controlPlane.platform** 参数值匹配。	**aws**、**azure**、**gcp**、**openstack**、**ovirt**、**vsphere** 或 **{}**
compute.replicas	要置备的计算机器数量，也称为 worker 机器。	大于或等于 **2** 的正整数。默认值为 **3**。
controlPlane	组成 control plane 的机器的配置。	**MachinePool** 对象的数组。详情请查看以下"Machine-pool"表。
controlPlane.architecture	决定池中机器的指令集合架构。目前不支持异构集群，因此所有池都必须指定相同的架构。有效值为 **amd64** （默认值）。	字符串
controlPlane.hyperthreading	是否在 control plane 机器上启用或禁用并发多线程或**超线程**。默认情况下，启用并发多线程以提高机器内核的性能。  **重要** 如果禁用并发多线程，请确保在容量规划时考虑到机器性能可能会显著降低的问题。	**Enabled** 或 **Disabled**
controlPlane.name	使用 **controlPlane** 时需要。机器池的名称。	**master**
controlPlane.platform	使用 **controlPlane** 时需要。使用此参数指定托管 control plane 机器的云供应商。此参数值必须与 **compute.platform** 参数值匹配。	**aws**、**azure**、**gcp**、**openstack**、**ovirt**、**vsphere** 或 **{}**
controlPlane.replicas	要置备的 control plane 机器数量。	唯一支持的值是 **3**，它是默认值。

参数	描述	值
credentialsMode	Cloud Credential Operator（CCO）模式。如果没有指定任何模式，CCO 会动态地尝试决定提供的凭证的功能，在支持多个模式的平台上使用 mint 模式。  **注意** 不是所有 CCO 模式都支持所有云供应商。如需有关 CCO 模式的更多信息，请参阅*Red Hat Operator 参考指南*内容中的 *Cloud Credential Operator* 条目。	**Mint**、**Passthrough**、**Manual** 或空字符串(**""**)。
fips	启用或禁用 FIPS 模式。默认为 **false**（禁用）。如果启用了 FIPS 模式，运行 OpenShift Container Platform 的 Red Hat Enterprise Linux CoreOS (RHCOS) 机器会绕过默认的 Kubernetes 加密套件，并使用由 RHCOS 提供的加密模块。  **重要** 只有在 **x86_64** 架构中的 OpenShift Container Platform 部署支持 FIPS 验证的/Modules in Process 加密库。  **注意** 如果使用 Azure File 存储，则无法启用 FIPS 模式。	**false** 或 **true**
imageContentSources	release-image 内容的源和仓库。	对象数组。包括一个 **source** 以及可选的 **mirrors**，如下表所示。
imageContentSources.source	使用 **imageContentSources** 时需要。指定用户在镜像拉取规格中引用的仓库。	字符串

参数	描述	值
imageContentSources.mirrors	指定可能还包含同一镜像的一个或多个仓库。	字符串数组
publish	如何发布或公开集群的面向用户的端点，如 Kubernetes API、OpenShift 路由。	**Internal** 或 **External**。把 **publish** 设置为 **Internal** 以部署一个私有集群，它不能被互联网访问。默认值为 **External**。
sshKey	用于验证集群机器访问的 SSH 密钥或密钥。  **注意** 对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 **ssh-agent** 进程使用的 SSH 密钥。	一个或多个密钥。例如： sshKey: <key1> <key2> <key3>

3.6.6.1.4. 其他 Azure 配置参数

下表描述了其他 Azure 配置参数：

表 3.18. 其他 Azure 参数

参数	描述	值
compute.platform.azure.osDisk.diskSizeGB	虚拟机的 Azure 磁盘大小。	以 GB 为单位表示磁盘大小的整数。默认值为 **128**。
compute.platform.azure.osDisk.diskType	定义磁盘的类型。	**Standard_LRS**、**Premium_LRS** 或 **标准SSD_LRS**。默认值为 **Premium_LRS**。
controlPlane.platform.azure.osDisk.diskSizeGB	虚拟机的 Azure 磁盘大小。	以 GB 为单位表示磁盘大小的整数。默认值为 **1024**。

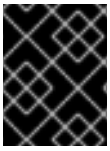
参数	描述	值
controlPlane.platform.azure.osDisk.diskType	定义磁盘的类型。	**Premium_LRS** 或 **标准SSD_LRS**. 默认值为 **Premium_LRS**。
platform.azure.baseDomainResourceGroupName	包含基域的 DNS 区的资源组的名称。	字符串，如 **production_cluster**。
platform.azure.outboundType	用于将集群连接到互联网的出站路由策略。如果使用用户定义的路由，则必须在安装集群前配置出站路由。安装程序不负责配置用户定义的路由。	**loadbalancer** 或 **UserDefinedRouting**。默认为 **LoadBalancer**。
platform.azure.region	托管集群的 Azure 区域名称。	任何有效的区域名称，如 **centralus**。
platform.azure.zone	可在其中放入机器的可用区的列表。如需高可用性，请至少指定两个区域。	区域列表，如 **["1", "2", "3"]**。
platform.azure.networkResourceGroupName	包含要将集群部署到的现有 VNet 的资源组名称。这个名称不能和 **platform.azure.baseDomainResourceGroupName**相同。	字符串。
platform.azure.virtualNetwork	要将集群部署到的现有 VNet 的名称。	字符串。
platform.azure.controlPlaneSubnet	要将 control plane 机器部署到的 VNet 中现有子网的名称。	有效的 CIDR，如 **10.0.0.0/16**。
platform.azure.computeSubnet	您要将计算机器部署到的 VNet 中现有子网的名称。	有效的 CIDR，如 **10.0.0.0/16**。
platform.azure.cloudName	用于使用适当 Azure API 端点配置 Azure SDK 的 Azure 云环境名称。如果为空，则使用默认值 **AzurePublicCloud**。	任何有效的云环境，如 **AzurePublicCloud** 或 **AzureUSGovernmentCloud**。

注意

您无法自定义 Azure 可用区，也不能使用标签来整理用于 Azure 集群的 Azure 资源。

3.6.6.2. Azure 的自定义 install-config.yaml 文件示例

您可以自定义 **install-config.yaml** 文件，以指定有关 OpenShift Container Platform 集群平台的更多信息，或修改所需参数的值。



重要

此示例 YAML 文件仅供参考。您必须使用安装程序来获取 **install-config.yaml** 文件，并且修改该文件。

```
apiVersion: v1
baseDomain: example.com (1)
controlPlane: (2)
  hyperthreading: Enabled (3) (4)
  name: master
  platform:
    azure:
      osDisk:
        diskSizeGB: 1024 (5)
        diskType: Premium_LRS
      type: Standard_D8s_v3
  replicas: 3
compute: (6)
- hyperthreading: Enabled (7)
  name: worker
  platform:
    azure:
      type: Standard_D2s_v3
      osDisk:
        diskSizeGB: 512 (8)
        diskType: Standard_LRS
      zones: (9)
      - "1"
      - "2"
      - "3"
  replicas: 5
metadata:
  name: test-cluster (10)
networking:
  clusterNetwork:
  - cidr: 10.128.0.0/14
    hostPrefix: 23
  machineNetwork:
  - cidr: 10.0.0.0/16
  networkType: OpenShiftSDN
  serviceNetwork:
  - 172.30.0.0/16
platform:
  azure:
    baseDomainResourceGroupName: resource_group (11)
    region: centralus (12)
    resourceGroupName: existing_resource_group (13)
    networkResourceGroupName: vnet_resource_group (14)
    virtualNetwork: vnet (15)
    controlPlaneSubnet: control_plane_subnet (16)
    computeSubnet: compute_subnet (17)
    outboundType: Loadbalancer
    cloudName: AzurePublicCloud
```

```
pullSecret: '{"auths": ...}' 18
fips: false 19
sshKey: ssh-ed25519 AAAA... 20
```

1 10 12 18 必需。安装程序会提示您输入这个值。

2 6 如果没有提供这些参数和值，安装程序会提供默认值。

3 7 **controlPlane** 部分是一个单映射，但 **compute** 部分是一系列映射。为满足不同数据结构的要求，**compute** 部分的第一行必须以连字符 **-** 开头，**controlPlane** 部分的第一行则不可以连字符开头。只使用一个 control plane 池。

4 是否要启用或禁用并发多线程或**超线程**。默认情况下，启用并发多线程以提高机器内核的性能。您可以通过将参数值设为 **Disabled** 来禁用。如果您在某些集群机器上禁用并发多线程，则必须在所有集群机器上禁用。

重要

如果禁用并发多线程，请确保在容量规划时考虑到机器性能可能会显著降低的问题。如果禁用并发多线程，请使用较大的虚拟机类型，如 **Standard_D8s_v3**。

5 8 可以 GB 为单位指定要使用的磁盘大小。control plane 节点（也称为 master 节点）的最低推荐值为 1024 GB。

9 指定要将机器部署到的区域列表。如需高可用性，请至少指定两个区域。

11 指定包含基域的 DNS 区的资源组的名称。

13 指定要安装集群的现有资源组的名称。如果未定义，则会为集群创建新的资源组。

14 如果您使用现有的 如果使用现有的 VNet，请指定包含它的资源组的名称。。

15 如果使用现有的 VNet，请指定其名称。

16 如果使用现有的 VNet，请指定托管 control plane 机器的子网名称。

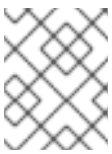
17 如果使用现有的 VNet，请指定托管计算机器的子网名称。

19 是否启用或禁用 FIPS 模式。默认情况下不启用 FIPS 模式。如果启用了 FIPS 模式，运行 OpenShift Container Platform 的 Red Hat Enterprise Linux CoreOS (RHCOS) 机器会绕过默认的 Kubernetes 加密套件，并使用由 RHCOS 提供的加密模块。

重要

只有在 **x86_64** 架构中的 OpenShift Container Platform 部署支持 FIPS 验证的/Modules in Process 加密库。

20 您可以选择提供您用来访问集群中机器的 **sshKey** 值。



注意

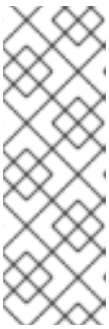
对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 **ssh-agent** 进程使用的 SSH 密钥。

3.6.6.3. 在安装过程中配置集群范围代理

生产环境可能会拒绝直接访问互联网，而是提供 HTTP 或 HTTPS 代理。您可以通过在 **install-config.yaml** 文件中配置代理设置，将新的 OpenShift Container Platform 集群配置为使用代理。

先决条件

- 您有一个现有的 **install-config.yaml** 文件。
- 您检查了集群需要访问的站点，并决定是否需要绕过代理。默认情况下代理所有集群出口流量，包括对托管云供应商 API 的调用。您需要将站点添加到 **Proxy** 对象的 **spec.noProxy** 字段来绕过代理。



注意

Proxy 对象 **status.noProxy** 字段使用安装配置中的 **networking.machineNetwork[].cidr**、**networking.clusterNetwork[].cidr** 和 **networking.serviceNetwork[]** 字段的值填充。

对于在 Amazon Web Services(AWS)、Google Cloud Platform(GCP)、Microsoft Azure 和 Red Hat OpenStack Platform(RHOSP)上安装, **Proxy** 对象 **status.noProxy** 字段也会使用实例元数据端点填充(**169.254.169.254**)。

流程

1. 编辑 **install-config.yaml** 文件并添加代理设置。例如：

```
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: example.com 3
additionalTrustBundle: | 4
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
...
```

1. 用于创建集群外 HTTP 连接的代理 URL。URL 必须是 **http**。
2. 用于创建集群外 HTTPS 连接的代理 URL。
3. 要排除在代理中的目标域名、IP 地址或其他网络 CIDR 的逗号分隔列表。在域前面加 **.** 来仅匹配子域。例如： **.y.com** 匹配 **x.y.com**，但不匹配 **y.com**。使用 ***** 绕过所有目的地的代理。
4. 如果提供，安装程序会在 **openshift-config 命名空间中生成名为 user-ca- bundle** 的配置映射来保存额外的 CA 证书。如果您提供 **additionalTrustBundle** 和至少一个代理设置，则 **Proxy** 对象会被配置为引用 **trustedCA** 字段中的 **user-ca-bundle** 配置映射。然后，Cluster Network Operator 会创建一个 **trusted-ca-bundle** 配置映射，该配置映射将为 **trustedCA** 参数指定的内容与 RHCOS 信任捆绑包合并。 **additionalTrustBundle** 字段是必需的，除非代理的身份证书由来自 RHCOS 信任捆绑包的颁发机构签名。

注意

安装程序不支持代理的 **readinessEndpoints** 字段。

2. 保存该文件，并在安装 OpenShift Container Platform 时引用。

安装程序会创建一个名为 **cluster** 的集群范围代理，该代理使用提供的 **install-config.yaml** 文件中的代理设置。如果没有提供代理设置，仍然会创建一个 **cluster Proxy** 对象，但它会有一个空 **spec**。

注意

只支持名为 **cluster** 的 **Proxy** 对象，且无法创建额外的代理。

3.6.7. 部署集群

您可以在兼容云平台中安装 OpenShift Container Platform。

重要

安装程序的 **create cluster** 命令只能在初始安装过程中运行一次。

先决条件

- 配置托管集群的云平台的帐户。
- 获取 OpenShift Container Platform 安装程序以及集群的 pull secret。

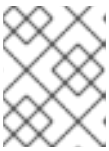
流程

1. 更改为包含安装程序的目录并初始化集群部署：

```
$ ./openshift-install create cluster --dir <installation_directory> \ 1
    --log-level=info 2
```

1 对于 **<installation_directory>**，请指定自定义 **./install-config.yaml** 文件的位置。

2 要查看不同的安装详情，请指定 **warn**、**debug** 或 **error**，而不要指定 **info**。



注意

如果您在主机上配置的云供应商帐户没有足够的权限来部署集群，安装过程将会停止，并且显示缺少的权限。

集群部署完成后，终端会显示访问集群的信息，包括指向其 Web 控制台的链接和 **kubeadmin** 用户的凭证。

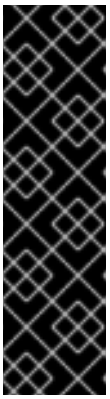
输出示例

```
...
INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run 'export
KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'
```

```
INFO Access the OpenShift web-console here: https://console-openshift-
console.apps.mycluster.example.com
INFO Login to the console with user: "kubeadmin", and password: "4vYBz-Ee6gm-ymBZj-
Wt5AL"
INFO Time elapsed: 36m22s
```

注意

当安装成功时，集群访问和凭证信息还会输出到 **<installation_directory>/.openshift_install.log**。



重要

- 安装程序生成的 Ignition 配置文件包含在 24 小时后过期的证书，然后在过期时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外情况是，您需要手动批准待处理的 **node-bootstrapper** 证书签名请求（CSR）来恢复 kubelet 证书。如需更多信息，请参阅*从过期的 control plane 证书中恢复*的文档。
- 建议您在生成 12 小时后使用 Ignition 配置文件，因为集群安装后 24 小时证书从 16 小时轮转至 22 小时。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中运行证书更新时避免安装失败。

重要

您不得删除安装程序或安装程序所创建的文件。需要这两者才能删除集群。

3.6.8. 通过下载二进制文件安装 OpenShift CLI

您需要安装 CLI（**oc**） 来使用命令行界面与 OpenShift Container Platform 进行交互。您可在 Linux 、Windows 或 macOS 上安装 **oc**。

重要

如果安装了旧版本的 **oc**，则无法使用 OpenShift Container Platform 4.6 中的所有命令。下载并安装新版本的 **oc**。

3.6.8.1. 在 Linux 上安装 OpenShift CLI

您可以按照以下流程在 Linux 上安装 OpenShift CLI（**oc**）二进制文件。

流程

1. 进入到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 在 **Version** 下拉菜单中选择相应的版本。
3. 单击 **OpenShift v4.6 Linux 客户端** 条目旁边的 **Download Now**，再保存文件。
4. 解包存档：

```
$ tar xvzf <file>
```

5. 把 **oc** 二进制代码放到 **PATH** 中的目录中。
   执行以下命令可以查看当前的 **PATH** 设置：

```
$ echo $PATH
```

安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
$ oc <command>
```

3.6.8.2. 在 Windows 上安装 OpenShift CLI

您可以按照以下流程在 Windows 上安装 OpenShift CLI（**oc**）二进制代码。

流程

1. 进入到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 在 **Version** 下拉菜单中选择相应的版本。
3. 单击 **OpenShift v4.6 Windows 客户端**条目旁边的 **Download Now**，再保存文件。
4. 使用 ZIP 程序解压存档。
5. 把 **oc** 二进制代码放到 **PATH** 中的目录中。
   要查看您的 **PATH**，请打开命令提示窗口并执行以下命令：

```
C:\> path
```

安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
C:\> oc <command>
```

3.6.8.3. 在 macOS 上安装 OpenShift CLI

您可以按照以下流程在 macOS 上安装 OpenShift CLI（**oc**）二进制代码。

流程

1. 进入到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 在 **Version** 下拉菜单中选择相应的版本。
3. 单击 **OpenShift v4.6 MacOSX 客户端**条目旁边的 **Download Now**，再保存文件。
4. 解包和解压存档。
5. 将 **oc** 二进制文件移到 PATH 的目录中。
   要查看您的 **PATH**，打开一个终端窗口并执行以下命令：

```
$ echo $PATH
```

安装 OpenShift CLI 后，可以使用 **oc** 命令：

-

```
$ oc <command>
```

3.6.9. 使用 CLI 登录到集群

您可以通过导出集群 **kubeconfig** 文件，以默认系统用户身份登录集群。**kubeconfig** 文件包含关于集群的信息，供 CLI 用于将客户端连接到正确集群和 API 服务器。该文件特只适用于一个特定的集群，在 OpenShift Container Platform 安装过程中创建。

先决条件

- 已部署了 OpenShift Container Platform 集群。
- 已安装 **oc** CLI。

流程

1. 导出 **kubeadmin** 凭证：

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig
```
(1)

(1) 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

2. 使用导出的配置，验证能否成功运行 **oc** 命令：

```
$ oc whoami
```

输出示例

```
system:admin
```

其他资源

- 如需有关访问和了解 OpenShift Container Platform Web 控制台的更多信息，请参阅访问 Web 控制台。

3.6.10. OpenShift Container Platform 的 Telemetry 访问

在 OpenShift Container Platform 4.6 中，默认运行的 Telemetry 服务提供有关集群健康状况和成功更新的指标，需要访问互联网。如果您的集群连接到互联网，Telemetry 会自动运行，而且集群会注册到 OpenShift Cluster Manager。

确认 OpenShift Cluster Manager 清单正确后，可以由 Telemetry 自动维护，也可以使用 OpenShift Cluster Manager 手动维护，使用订阅监控来跟踪帐户或多集群级别的 OpenShift Container Platform 订阅。

其他资源

- 有关 Telemetry 服务的更多信息，请参阅关于远程健康监控。

3.6.11. 后续步骤

- 自定义集群。
- 若有需要，您可以选择不使用远程健康报告。

3.7. 在 AZURE 上安装私有集群

在 OpenShift Container Platform 版本 4.6 中，您可以在 Microsoft Azure 上将私有集群安装到现有 Azure Virtual Network（VNet）中。安装程序会置备所需基础架构的其余部分，您可以进一步定制这些基础架构。要自定义安装，请在安装集群前修改 **install-config.yaml** 文件中的参数。

3.7.1. 先决条件

- 查看有关 OpenShift Container Platform 安装和更新流程的详细信息。
- 配置一个 Azure 帐户以托管集群，并决定要将集群部署到的已测试和验证的区域。
- 如果使用防火墙，则必须将其配置为允许集群需要访问的站点。
- 如果不允许系统管理身份和访问管理（IAM），集群管理员可以 手动创建和维护 IAM 凭证。手动模式也可以用于云 IAM API 无法访问的环境中。

3.7.2. 私有集群

您可以部署不公开外部端点的私有 OpenShift Container Platform 集群。私有集群只能从内部网络访问，且无法在互联网中看到。

默认情况下，OpenShift Container Platform 被置备为使用可公开访问的 DNS 和端点。私有集群在部署集群时将 DNS、Ingress Controller 和 API 服务器设置为私有。这意味着，集群资源只能从您的内部网络访问，且不能在互联网中看到。

要部署私有集群，您必须使用符合您的要求的现有网络。您的集群资源可能会在网络中的其他集群间共享。

另外，您必须从可访问您置备的云的 API 服务、您置备的网络上的主机以及可以连接到互联网来获取安装介质的机器上部署私有集群。您可以使用符合这些访问要求的机器，并按照您的公司规定进行操作。例如，该机器可以是云网络中的堡垒主机，也可以是可通过 VPN 访问网络的机器。

3.7.2.1. Azure 中的私有集群

要在 Microsoft Azure 平台上创建私有集群，您必须提供一个现有的私有 VNet 和子网来托管集群。安装程序还必须能够解析集群所需的 DNS 记录。安装程序只为内部流量配置 Ingress Operator 和 API 服务器。

根据您的网络如何连接到私有 VNET，您可能需要使用 DNS 转发程序来解析集群的专用 DNS 记录。集群的机器在内部使用 **168.63.129.16** 进行 DNS 解析。如需更多信息，请参阅Azure 文档中的 Azure Private DNS？ 和 什么是 IP 地址 168.63.129.16？ 部分 。

集群仍然需要访问互联网来访问 Azure API。

安装私有集群时不需要或创建以下项目：

- **BaseDomainResourceGroup**，因为集群不创建公共记录
- 公共 IP 地址

- 公共 DNS 记录
- 公共端点

> The cluster is configured so that the Operators do not create public records for the cluster and all cluster machines are placed in the private subnets that you specify.

3.7.2.1.1. 限制：

Azure 上的私有集群只受到与使用一个现有 VNet 相关的限制。

3.7.2.2. 用户定义的出站路由

在 OpenShift Container Platform 中，您可以选择自己的出站路由来连接到互联网。这可让您跳过创建公共 IP 地址和公共负载均衡器的步骤。

您可在安装集群前修改 **install-config.yaml** 文件中的参数来配置用户定义的路由。安装集群时，需要一个已存在的 VNet 来使用出站路由，安装程序不负责配置它。

当将集群配置为使用用户定义的路由时，安装程序不会创建以下资源：

- 用于访问互联网的出站规则。
- 公共负载均衡器的公共 IP。
- Kubernetes Service 对象，为出站请求将集群机器添加到公共负载均衡器中。

在设置用户定义的路由前，您必须确保以下项目可用：

- 出口到互联网可以拉取容器镜像，除非使用内部 registry 镜像。
- 集群可以访问 Azure API。
- 配置了各种允许列表端点。您可以在*配置防火墙*部分引用这些端点。

支持一些已存在的网络设置，使用用户定义的路由访问互联网。

带有网络地址转换的专用集群
您可以使用 Azure VNET 网络地址转换(NAT) 为集群中的子网提供出站互联网访问。请参阅 Azure 文档中的使用 Azure CLI 创建 NAT 网关 部分。

当使用配置了 Azure NAT 和用户定义的路由的 VNet 设置时，您可以创建没有公共端点的私有集群。

使用 Azure 防火墙的私有集群
您可以使用 Azure Firewall 为用于安装集群的 VNet 提供出站路由。请参阅 Azure 文档中的Azure Firewall 提供用户定义的路由的信息。

使用 Azure Firewall 和用户自定义路由的 VNet 设置时，您可以创建没有公共端点的私有集群。

带有代理配置的私有集群
您可以使用带有用户定义的路由的代理来允许到互联网的出口。您必须确保集群 Operator 不使用代理访问 Azure API。Operator 必须有权访问代理外的 Azure API。

当使用子网的默认路由表时，Azure 会自动填充 **0.0.0.0/0**，所有 Azure API 请求也会通过 Azure 的内部网络路由，即使 IP 地址是公共的。只要网络安全组规则允许出口到 Azure API 端点，您就可以在没有公共端点的情况下创建用户自定义路由的代理。

没有互联网访问的私有集群
您可以安装专用网络，以限制对互联网的所有访问，但 Azure API 除外。这可以通过在本地镜像发行版本镜像 registry 实现。集群必须有权访问以下内容：

- 可以从中拉取容器镜像的内部 registry 镜像
- 访问 Azure API

在满足这些要求时，就可以使用用户定义的路由来创建没有公共端点的私有集群。

3.7.3. 关于为 OpenShift Container Platform 集群重复使用 VNet

在 OpenShift Container Platform 4.6 中，您可以在 Microsoft Azure 中将集群部署到现有的 Azure Virtual Network（VNet）中。如果您这样做，还必须在 VNet 和路由规则中使用现有子网。

通过将 OpenShift Container Platform 部署到现有的 Azure VNet 中，您可能会避开新帐户中的服务限制，或者更容易地利用公司所设置的操作限制。如果您无法获得创建 VNet 所需的基础架构创建权限，则可以使用这个选项。

3.7.3.1. 使用 VNet 的要求

当使用现有 VNet 部署集群时，必须在安装集群前执行额外网络配置。在安装程序置备的基础架构集群中，安装程序通常会创建以下组件，但在安装到现有 VNet 时不会创建它们：

- 子网
- 路由表
- VNets
- 网络安全组

注意

安装程序要求您使用由云提供的 DNS 服务器。不支持使用自定义 DNS 服务器，并导致安装失败。

如果您使用自定义 VNet，您必须正确配置它及其子网，以便安装程序和集群使用。安装程序不能为集群分配要使用的网络范围，为子网设置路由表，或者设置类似 DHCP 的 VNet 选项，因此您必须在安装集群前配置它们。

集群必须能够访问包含现有 VNet 和子网的资源组。虽然集群创建的所有资源都放在它创建的单独资源组中，但有些网络资源则从另外一个独立的组中使用。一些集群 Operator 必须能够访问这两个资源组中的资源。例如，Machine API 控制器会为它创建的虚拟机附加 NICS，使其从网络资源组中划分子网。

您的 VNet 必须满足以下特征：

- VNet 的 CIDR 块必须包含 **Networking.machineCIDR**，它是集群机器的 IP 地址池。
- VNet 及其子网必须属于同一资源组，子网必须配置为使用 Azure 分配的 DHCP IP 地址而不是静态 IP 地址。

您必须在 VNet 中提供两个子网，一个用于 control plane 机器，一个用于计算机器。因为 Azure 在您指定的区域内的不同可用区中分发机器，所以集群将默认具有高可用性功能。

要确保您提供的子网适合您的环境，安装程序会确认以下信息：

- 所有指定的子网都存在。
- 有两个专用子网，一个用于 control plane 机器，一个用于计算机器。
- 子网 CIDR 属于您指定的机器 CIDR。机器不会在没有为其提供私有子网的可用区中置备。

注意

如果您销毁了使用现有 VNet 的集群，则不会删除 VNet。

3.7.3.1.1. 网络安全组要求

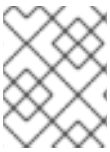
托管 compute 和 control plane 机器的子网的网络安全组需要特定的访问权限，以确保集群通信正确。您必须创建规则来允许访问所需的集群通信端口。

重要

在安装集群前必须先设置网络安全组规则。如果您试图在没有所需访问权限的情况下安装集群，安装程序就无法访问 Azure API，且会导致安装失败。

表 3.19. 所需端口

端口	描述	Control plane	Compute
80	允许 HTTP 流量		x
443	允许 HTTPS 流量		x
6443	允许与 control plane 机器通信。	x	
22623	允许与机器配置服务器通信。	x	



注意

由于集群组件不会修改用户提供的网络安全组（Kubernetes 控制器更新它），因此会创建一个伪网络安全组，供 Kubernetes 控制器修改，而不影响其余的环境。

3.7.3.2. 权限划分

从 OpenShift Container Platform 4.3 开始，您不需要安装程序置备的基础架构集群部署所需的所有权限。这与您所在机构可能已有的权限划分类似：不同的个人可以在您的云中创建不同的资源。例如，您可以创建针对于特定应用程序的对象，如实例、存储和负载均衡器，但不能创建与网络相关的组件，如 VNets 、子网或入站规则。

您在创建集群时使用的 Azure 凭证不需要 VNets 和核心网络组件（如子网、路由表、互联网网关、NAT 和 VPN）所需的网络权限。您仍然需要获取集群中的机器需要的应用程序资源的权限，如负载均衡器 、安全组、存储帐户和节点。

3.7.3.3. 集群间隔离

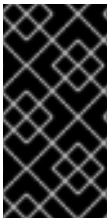
因为集群无法修改现有子网中的网络安全组，所以无法在 VNet 中相互隔离集群。

3.7.4. OpenShift Container Platform 的互联网访问

在 OpenShift Container Platform 4.6 中，您需要访问互联网来安装集群。

您必须具有以下互联网访问权限：

- 访问 OpenShift Cluster Manager 以下载安装程序并执行订阅管理。如果集群可以访问互联网，并且没有禁用 Telemetry，该服务会自动授权您的集群。
- 访问 Quay.io，以获取安装集群所需的软件包。
- 获取执行集群更新所需的软件包。



重要

如果您的集群无法直接访问互联网，则可以在置备的某些类基础架构上执行受限网络安装。在此过程中，您要下载所需的内容，并使用它在镜像 registry（mirror registry） 中填充安装集群并生成安装程序所需的软件包。对于某些安装类型，集群要安装到的环境不需要访问互联网。在更新集群之前，要更新 registry 镜像系统中的内容。

3.7.5. 生成 SSH 私钥并将其添加到代理中

如果要在集群上执行安装调试或灾难恢复，则必须为 **ssh-agent** 和安装程序提供 SSH 密钥。您可以使用此密钥访问公共集群中的 bootstrap 机器来排除安装问题。

注意

在生产环境中，您需要进行灾难恢复和调试。

您可以使用此密钥以 **core** 用户身份通过 SSH 连接到 master 节点。在部署集群时，此密钥会添加到 **core** 用户的 **~/.ssh/authorized_keys** 列表中。

注意

您必须使用一个本地密钥，而不要使用在特定平台上配置的密钥，如 AWS 密钥对。

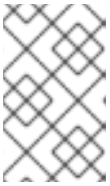
流程

1. 如果还没有为计算机上免密码身份验证而配置的 SSH 密钥，请创建一个。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ ssh-keygen -t ed25519 -N '' \
  -f <path>/<file_name> 1
```

1 指定新 SSH 密钥的路径和文件名，如 **~/.ssh/id_rsa**。如果您已有密钥对，请确保您的公钥位于 **~/.ssh** 目录中。

运行此命令会在指定的位置生成不需要密码的 SSH 密钥。



注意

如果您计划在 **x86_64** 架构中安装使用 FIPS 验证的/Modules in Process 加密库的 OpenShift Container Platform 集群，不要创建使用 **ed25519** 算法的密钥。反之，创建一个使用 **rsa** 或 **ecdsa** 算法的密钥。

2. 作为后台任务启动 **ssh-agent** 进程：

```
$ eval "$(ssh-agent -s)"
```

输出示例

```
Agent pid 31874
```

注意

如果您的集群采用 FIPS 模式，则只使用 FIPS 兼容算法来生成 SSH 密钥。密钥必须是 RSA 或 ECDSA。

3. 将 SSH 私钥添加到 **ssh-agent**：

```
$ ssh-add <path>/<file_name> 1
```

输出示例

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

1 指定 SSH 私钥的路径和文件名，如 **~/.ssh/id_rsa**

后续步骤

- 在安装 OpenShift Container Platform 时，为安装程序提供 SSH 公钥。

3.7.6. 获取安装程序

在安装 OpenShift Container Platform 之前，将安装文件下载到本地计算机上。

先决条件

- 运行 Linux 或 macOS 的计算机，本地磁盘空间为 500 MB

流程

1. 访问 OpenShift Cluster Manager 站点的 Infrastructure Provider 页面。如果您有红帽帐号，请使用自己的凭证登录。如果没有，请创建一个帐户。
2. 选择您的基础架构供应商。
3. 进入适用于您的安装类型的页面，下载您的操作系统的安装程序，并将文件放在要保存安装配置文件的目录中。。

重要

安装程序会在用来安装集群的计算机上创建若干文件。在完成集群安装后，您必须保留安装程序和安装程序所创建的文件。这两个文件都需要删除集群。

重要

删除安装程序创建的文件不会删除您的集群，即使集群在安装过程中失败也是如此。要删除集群，为特定云供应商完成 OpenShift Container Platform 卸载流程。

4. 提取安装程序。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ tar xvf openshift-install-linux.tar.gz
```

5. 从 Red Hat OpenShift Cluster Manager 下载安装 pull secret 。通过此 pull secret，您可以进行所含授权机构提供的服务的身份验证，这些服务包括为 OpenShift Container Platform 组件提供容器镜像的 Quay.io。

3.7.7. 手动创建安装配置文件

对于只能从内部网络访问且不能在互联网中看到的私有 OpenShift Container Platform 集群安装，您必须手动生成安装配置文件。

先决条件

- 获取 OpenShift Container Platform 安装程序和集群的访问令牌。

流程

1. 创建用来存储您所需的安装资产的安装目录：

```
$ mkdir <installation_directory>
```

重要

您必须创建目录。一些安装信息，如 bootstrap X.509 证书，有较短的过期间隔，因此不要重复使用安装目录。如果要重复使用另一个集群安装中的个别文件，可以将其复制到您的目录中。但是，一些安装数据的文件名可能会在发行版本之间有所改变。从 OpenShift Container Platform 老版本中复制安装文件时要格外小心。

2. 自定义以下 **install-config.yaml** 文件模板，并将它保存到 **<installation_directory>** 中。

注意

此配置文件必须命名为 **install-config.yaml**。

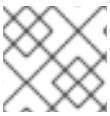
3. 备份 **install-config.yaml** 文件，以便用于安装多个集群。

重要

install-config.yaml 文件会在安装过程的下一步骤中消耗掉。现在必须备份它。

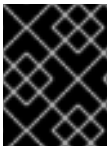
3.7.7.1. 安装配置参数

在部署 OpenShift Container Platform 集群前，您可以提供参数值，以描述托管集群的云平台的帐户并选择性地自定义集群平台。在创建 **install-config.yaml** 安装配置文件时，您可以通过命令行来提供所需的参数的值。如果要自定义集群，可以修改 **install-config.yaml** 文件来提供关于平台的更多信息。



注意

安装之后，您无法修改 **install-config.yaml** 文件中的这些参数。



重要

openshift-install 命令不验证参数的字段名称。如果指定了不正确的名称，则不会创建相关的文件或对象，且不会报告错误。确保所有指定的参数的字段名称都正确。

3.7.7.1.1. 所需的配置参数

下表描述了所需的安装配置参数：

表 3.20. 所需的参数

参数	描述	值
apiVersion	**install-config.yaml** 内容的 API 版本。当前版本是 **v1**。安装程序还可能支持旧的 API 版本。	字符串
baseDomain	云供应商的基域。此基础域用于创建到 OpenShift Container Platform 集群组件的路由。集群的完整 DNS 名称是 **baseDomain** 和 **metadata.name** 参数值的组合，其格式为 **<metadata.name>.<baseDomain>**。	完全限定域名或子域名，如 **example.com**。
metadata	Kubernetes 资源 **ObjectMeta**，其中只消耗 **name** 参数。	对象
metadata.name	集群的名称。集群的 DNS 记录是 **{{.metadata.name}}.{{.baseDomain}}** 的子域。	小写字母,连字符(-)和句点(.)的字符串，如 **dev**。
platform	执行安装的具体平台配置：**aws**、**baremetal**、**azure**、**openstack**、**ovirt**、**vsphere**。有关 **platform.<platform>** 参数的额外信息，请参考下表来了解您的具体平台。	对象

参数	描述	值

pullSecret	从 Red Hat OpenShift Cluster Manager 获取 pull secret，验证从 Quay.io 等服务中下载 OpenShift Container Platform 组件的容器镜像。	<pre>{
		  "auths":{
		  "cloud.openshift.com":{
		  "auth":"b3Blb=",
		  "email":"you@example.com"
		  },
		  "quay.io":{
		  "auth":"b3Blb=",
		  "email":"you@example.com"
		  }
		  }
		}</pre>

3.7.7.1.2. 网络配置参数

您可以根据现有网络基础架构的要求自定义安装配置。例如，您可以扩展集群网络的 IP 地址块，或者提供不同于默认值的不同 IP 地址块。

只支持 IPv4 地址。

表 3.21. 网络参数

参数	描述	值
networking	集群网络的配置。	对象 **注意** 您不能在安装后修改 **networking** 对象指定的参数。
networking.networkType	要安装的集群网络供应商 Container Network Interface（CNI）插件。	**OpenShiftSDN** 或 **OVNKubernetes**。默认值为 **OpenShiftSDN**。

networking.clusterNetwork	pod 的 IP 地址块。
	
	默认值为 **10.128.0.0/14**，主机前缀为 **/23**。
	
	如果您指定多个 IP 地址块，则块不得互相重叠。
		一个对象数组。例如：
		
		<pre>networking:
		  clusterNetwork:
		  - cidr: 10.128.0.0/14
		  hostPrefix: 23</pre>

networking.clusterNetwork.cidr	使用 **networking.clusterNetwork** 时需要此项。IP 地址块。 一个 IPv4 网络。	使用 CIDR 形式的 IP 地址块。IPv4 块的前缀长度介于 **0** 到 **32** 之间。

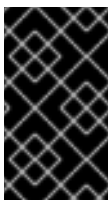
参数	描述	值
networking.clusterNetwork.hostPrefix	分配给每个单独节点的子网前缀长度。例如，如果 **hostPrefix** 设为 **23**，则每个节点从所给的 **cidr** 中分配一个 **/23** 子网。**hostPrefix** 值 **23** 提供 510（2^(32 - 23)- 2）个 pod IP 地址。	子网前缀。 默认值为 **23**。
networking.serviceNetwork	服务的 IP 地址块。默认值为 **172.30.0.0/16**。 OpenShift SDN 和 OVN-Kubernetes 网络供应商只支持服务网络的一个 IP 地址块。	CIDR 格式具有 IP 地址块的数组。例如： networking: serviceNetwork: - 172.30.0.0/16
networking.machineNetwork	机器的 IP 地址块。 如果您指定多个 IP 地址块，则块不得互相重叠。	一个对象数组。例如： networking: machineNetwork: - cidr: 10.0.0.0/16
networking.machineNetwork.cidr	使用 **networking.machineNetwork** 时需要。IP 地址块。libvirt 以外的所有平台的默认值为 **10.0.0.0/16**。对于 libvirt，默认值为 **192.168.126.0/24**。	CIDR 表示法中的 IP 网络块。 例如： **10.0.0.0/16**。 **注意** 将 **networking.machineNetwork** 设置为与首选 NIC 所在的 CIDR 匹配。

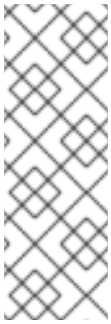
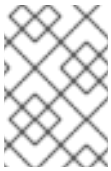
3.7.7.1.3. 可选配置参数

下表描述了可选安装配置参数：

表 3.22. 可选参数

参数	描述	值
additionalTrustBundle	添加到节点可信证书存储中的 PEM 编码 X.509 证书捆绑包。配置了代理时，也可以使用这个信任捆绑包。	字符串
compute	组成计算节点的机器的配置。	machine-pool 对象的数组。详情请查看以下"Machine-pool"表。

参数	描述	值
compute.architecture	决定池中机器的指令集合架构。目前不支持异构集群，因此所有池都必须指定相同的架构。有效值为 **amd64** （默认值）。	字符串
compute.hyperthreading	是否在计算机器上启用或禁用并发多线程或**超线程**。默认情况下，启用并发多线程以提高机器内核的性能。  **重要** 如果禁用并发多线程，请确保在容量规划时考虑到机器性能可能会显著降低的问题。	**Enabled** 或 **Disabled**
compute.name	使用 **compute** 时需要此值。机器池的名称。	**worker**
compute.platform	使用 **compute** 时需要此值。使用此参数指定托管 worker 机器的云供应商。此参数值必须与 **controlPlane.platform** 参数值匹配。	**aws**、**azure**、**gcp**、**openstack**、**ovirt**、**vsphere** 或 **{}**
compute.replicas	要置备的计算机器数量，也称为 worker 机器。	大于或等于 **2** 的正整数。默认值为 **3**。
controlPlane	组成 control plane 的机器的配置。	**MachinePool** 对象的数组。详情请查看以下"Machine-pool"表。
controlPlane.architecture	决定池中机器的指令集合架构。目前不支持异构集群，因此所有池都必须指定相同的架构。有效值为 **amd64** （默认值）。	字符串
controlPlane.hyperthreading	是否在 control plane 机器上启用或禁用并发多线程或**超线程**。默认情况下，启用并发多线程以提高机器内核的性能。  **重要** 如果禁用并发多线程，请确保在容量规划时考虑到机器性能可能会显著降低的问题。	**Enabled** 或 **Disabled**

参数	描述	值
controlPlane.name	使用 **controlPlane** 时需要。机器池的名称。	**master**
controlPlane.platform	使用 **controlPlane** 时需要。使用此参数指定托管 control plane 机器的云供应商。此参数值必须与 **compute.platform** 参数值匹配。	**aws**、**azure**、**gcp**、**openstack**、**ovirt**、**vsphere** 或 **{}**
controlPlane.replicas	要置备的 control plane 机器数量。	唯一支持的值是 **3**，它是默认值。
credentialsMode	Cloud Credential Operator（CCO）模式。如果没有指定任何模式，CCO 会动态地尝试决定提供的凭证的功能，在支持多个模式的平台上使用 mint 模式。  **注意** 不是所有 CCO 模式都支持所有云供应商。如需有关 CCO 模式的更多信息，请参阅*Red Hat Operator 参考指南*内容中的 *Cloud Credential Operator* 条目。	**Mint**、**Passthrough**、**Manual** 或空字符串(**""**)。
fips	启用或禁用 FIPS 模式。默认为 **false**（禁用）。如果启用了 FIPS 模式，运行 OpenShift Container Platform 的 Red Hat Enterprise Linux CoreOS (RHCOS) 机器会绕过默认的 Kubernetes 加密套件，并使用由 RHCOS 提供的加密模块。  **重要** 只有在 **x86_64** 架构中的 OpenShift Container Platform 部署支持 FIPS 验证的/Modules in Process 加密库。  **注意** 如果使用 Azure File 存储，则无法启用 FIPS 模式。	**false** 或 **true**

参数	描述	值
imageContentSources	release-image 内容的源和仓库。	对象数组。包括一个 **source** 以及可选的 **mirrors**，如下表所示。
imageContentSources.source	使用 **imageContentSources** 时需要。指定用户在镜像拉取规格中引用的仓库。	字符串
imageContentSources.mirrors	指定可能还包含同一镜像的一个或多个仓库。	字符串数组
publish	如何发布或公开集群的面向用户的端点，如 Kubernetes API、OpenShift 路由。	**Internal** 或 **External**。把 **publish** 设置为 **Internal** 以部署一个私有集群，它不能被互联网访问。默认值为 **External**。
sshKey	用于验证集群机器访问的 SSH 密钥或密钥。 **注意** 对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 **ssh-agent** 进程使用的 SSH 密钥。	一个或多个密钥。例如： sshKey: <key1> <key2> <key3>

3.7.7.1.4. 其他 Azure 配置参数

下表描述了其他 Azure 配置参数：

表 3.23. 其他 Azure 参数

参数	描述	值
compute.platform.azure.osDisk.diskSizeGB	虚拟机的 Azure 磁盘大小。	以 GB 为单位表示磁盘大小的整数。默认值为 **128**。
compute.platform.azure.osDisk.diskType	定义磁盘的类型。	**Standard_LRS**、**Premium_LRS** 或 **标准SSD_LRS**。默认值为 **Premium_LRS**。
controlPlane.platform.azure.osDisk.diskSizeGB	虚拟机的 Azure 磁盘大小。	以 GB 为单位表示磁盘大小的整数。默认值为 **1024**。

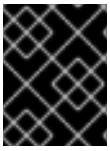
参数	描述	值
controlPlane.platform.azure.osDisk.diskType	定义磁盘的类型。	**Premium_LRS** 或 **标准SSD_LRS**. 默认值为 **Premium_LRS**。
platform.azure.baseDomainResourceGroupName	包含基域的 DNS 区的资源组的名称。	字符串，如 **production_cluster**。
platform.azure.outboundType	用于将集群连接到互联网的出站路由策略。如果使用用户定义的路由，则必须在安装集群前配置出站路由。安装程序不负责配置用户定义的路由。	**loadbalancer** 或 **UserDefinedRouting**。默认为 **LoadBalancer**。
platform.azure.region	托管集群的 Azure 区域名称。	任何有效的区域名称，如 **centralus**。
platform.azure.zone	可在其中放入机器的可用区的列表。如需高可用性，请至少指定两个区域。	区域列表，如 **["1", "2", "3"]**。
platform.azure.networkResourceGroupName	包含要将集群部署到的现有 VNet 的资源组名称。这个名称不能和 **platform.azure.baseDomainResourceGroupName**相同。	字符串。
platform.azure.virtualNetwork	要将集群部署到的现有 VNet 的名称。	字符串。
platform.azure.controlPlaneSubnet	要将 control plane 机器部署到的 VNet 中现有子网的名称。	有效的 CIDR，如 **10.0.0.0/16**。
platform.azure.computeSubnet	您要将计算机器部署到的 VNet 中现有子网的名称。	有效的 CIDR，如 **10.0.0.0/16**。
platform.azure.cloudName	用于使用适当 Azure API 端点配置 Azure SDK 的 Azure 云环境名称。如果为空，则使用默认值 **AzurePublicCloud**。	任何有效的云环境，如 **AzurePublicCloud** 或 **AzureUSGovernmentCloud**。

注意

您无法自定义 Azure 可用区，也不能使用标签来整理用于 Azure 集群的 Azure 资源。

3.7.7.2. Azure 的自定义 install-config.yaml 文件示例

您可以自定义 **install-config.yaml** 文件，以指定有关 OpenShift Container Platform 集群平台的更多信息，或修改所需参数的值。



重要

此示例 YAML 文件仅供参考。您必须使用安装程序来获取 **install-config.yaml** 文件，并且修改该文件。

```
apiVersion: v1
baseDomain: example.com (1)
controlPlane: (2)
  hyperthreading: Enabled (3) (4)
  name: master
  platform:
    azure:
      osDisk:
        diskSizeGB: 1024 (5)
        diskType: Premium_LRS
      type: Standard_D8s_v3
  replicas: 3
compute: (6)
- hyperthreading: Enabled (7)
  name: worker
  platform:
    azure:
      type: Standard_D2s_v3
      osDisk:
        diskSizeGB: 512 (8)
        diskType: Standard_LRS
      zones: (9)
      - "1"
      - "2"
      - "3"
  replicas: 5
metadata:
  name: test-cluster (10)
networking:
  clusterNetwork:
  - cidr: 10.128.0.0/14
    hostPrefix: 23
  machineNetwork:
  - cidr: 10.0.0.0/16
  networkType: OpenShiftSDN
  serviceNetwork:
  - 172.30.0.0/16
platform:
  azure:
    baseDomainResourceGroupName: resource_group (11)
    region: centralus (12)
    resourceGroupName: existing_resource_group (13)
    networkResourceGroupName: vnet_resource_group (14)
    virtualNetwork: vnet (15)
    controlPlaneSubnet: control_plane_subnet (16)
    computeSubnet: compute_subnet (17)
    outboundType: UserDefinedRouting (18)
    cloudName: AzurePublicCloud
```

```
pullSecret: '{"auths": ...}' 19
fips: false 20
sshKey: ssh-ed25519 AAAA... 21
publish: Internal 22
```

1 10 12 19 必需。安装程序会提示您输入这个值。

2 6 如果没有提供这些参数和值，安装程序会提供默认值。

3 7 **controlPlane** 部分是一个单映射，但 **compute** 部分是一系列映射。为满足不同数据结构的要求，**compute** 部分的第一行必须以连字符 **-** 开头，**controlPlane** 部分的第一行则不可以连字符开头。只使用一个 control plane 池。

4 是否要启用或禁用并发多线程或**超线程**。默认情况下，启用并发多线程以提高机器内核的性能。您可以通过将参数值设为 **Disabled** 来禁用。如果您在某些集群机器上禁用并发多线程，则必须在所有集群机器上禁用。

重要

如果禁用并发多线程，请确保在容量规划时考虑到机器性能可能会显著降低的问题。如果禁用并发多线程，请使用较大的虚拟机类型，如 **Standard_D8s_v3**。

5 8 可以 GB 为单位指定要使用的磁盘大小。control plane 节点（也称为 master 节点）的最低推荐值为 1024 GB。

9 指定要将机器部署到的区域列表。如需高可用性，请至少指定两个区域。

11 指定包含基域的 DNS 区的资源组的名称。

13 指定要安装集群的现有资源组的名称。如果未定义，则会为集群创建新的资源组。

14 如果您使用现有的 如果使用现有的 VNet，请指定包含它的资源组的名称。。

15 如果使用现有的 VNet，请指定其名称。

16 如果使用现有的 VNet，请指定托管 control plane 机器的子网名称。

17 如果使用现有的 VNet，请指定托管计算机器的子网名称。

18 您可以自定义自己的出站路由。配置用户定义的路由可防止在集群中公开外部端点。出口的用户自定义路由需要将集群部署到现有的 VNet。

20 是否启用或禁用 FIPS 模式。默认情况下不启用 FIPS 模式。如果启用了 FIPS 模式，运行 OpenShift Container Platform 的 Red Hat Enterprise Linux CoreOS (RHCOS) 机器会绕过默认的 Kubernetes 加密套件，并使用由 RHCOS 提供的加密模块。

重要

只有在 **x86_64** 架构中的 OpenShift Container Platform 部署支持 FIPS 验证的/Modules in Process 加密库。

21 您可以选择提供您用来访问集群中机器的 **sshKey** 值。

注意

对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 **ssh-agent** 进程使用的 SSH 密钥。

22 如何发布集群的面向用户的端点。把 **publish** 设置为 **Internal** 以部署一个私有集群，它不能被互联网访问。默认值为 **External**。

3.7.7.3. 在安装过程中配置集群范围代理

生产环境可能会拒绝直接访问互联网，而是提供 HTTP 或 HTTPS 代理。您可以通过在 **install-config.yaml** 文件中配置代理设置，将新的 OpenShift Container Platform 集群配置为使用代理。

先决条件

- 您有一个现有的 **install-config.yaml** 文件。
- 您检查了集群需要访问的站点，并决定是否需要绕过代理。默认情况下代理所有集群出口流量，包括对托管云供应商 API 的调用。您需要将站点添加到 **Proxy** 对象的 **spec.noProxy** 字段来绕过代理。

注意

Proxy 对象 **status.noProxy** 字段使用安装配置中的 **networking.machineNetwork[].cidr**、**networking.clusterNetwork[].cidr** 和 **networking.serviceNetwork[]** 字段的值填充。

对于在 Amazon Web Services(AWS)、Google Cloud Platform(GCP)、Microsoft Azure 和 Red Hat OpenStack Platform(RHOSP)上安装，**Proxy** 对象 **status.noProxy** 字段也会使用实例元数据端点填充(**169.254.169.254**)。

流程

1. 编辑 **install-config.yaml** 文件并添加代理设置。例如：

```
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: example.com 3
additionalTrustBundle: | 4
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
...
```

1 用于创建集群外 HTTP 连接的代理 URL。URL 必须是 **http**。

2 用于创建集群外 HTTPS 连接的代理 URL。

3 要排除在代理中的目标域名、IP 地址或其他网络 CIDR 的逗号分隔列表。在域前面加 **.** 来仅匹配子域。例如：**.y.com** 匹配 **x.y.com**，但不匹配 **y.com**。使用 ***** 绕过所有目的地的代理。

4 如果提供，安装程序会在 **openshift-config 命名空间中生成名为 user-ca- bundle** 的配置映射来保存额外的 CA 证书。如果您提供 **additionalTrustBundle** 和至少一个代理设置，则 **Proxy** 对象会被配置为引用 **trustedCA** 字段中的 **user-ca-bundle** 配置映射。然后，Cluster Network Operator 会创建一个 **trusted-ca-bundle** 配置映射，该配置映射将为 **trustedCA** 参数指定的内容与 RHCOS 信任捆绑包合并。**additionalTrustBundle** 字段是必需的，除非代理的身份证书由来自 RHCOS 信任捆绑包的颁发机构签名。

注意

安装程序不支持代理的 **readinessEndpoints** 字段。

2. 保存该文件，并在安装 OpenShift Container Platform 时引用。

安装程序会创建一个名为 **cluster** 的集群范围代理，该代理使用提供的 **install-config.yaml** 文件中的代理设置。如果没有提供代理设置，仍然会创建一个 **cluster Proxy** 对象，但它会有一个空 **spec**。

注意

只支持名为 **cluster** 的 **Proxy** 对象，且无法创建额外的代理。

3.7.8. 部署集群

您可以在兼容云平台中安装 OpenShift Container Platform。

重要

安装程序的 **create cluster** 命令只能在初始安装过程中运行一次。

先决条件

- 配置托管集群的云平台的帐户。
- 获取 OpenShift Container Platform 安装程序以及集群的 pull secret。

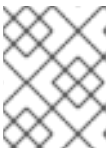
流程

1. 更改为包含安装程序的目录并初始化集群部署：

```
$ ./openshift-install create cluster --dir <installation_directory> \ 1
    --log-level=info 2
```

1 对于 **<installation_directory>**，请指定

2 要查看不同的安装详情，请指定 **warn**、**debug** 或 **error**，而不要指定 **info**。



注意

如果您在主机上配置的云供应商帐户没有足够的权限来部署集群，安装过程将会停止，并且显示缺少的权限。

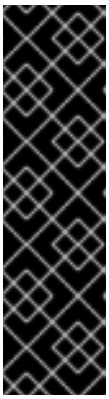
集群部署完成后，终端会显示访问集群的信息，包括指向其 Web 控制台的链接和 **kubeadmin** 用户的凭证。

输出示例

```
...
INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run 'export
KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'
INFO Access the OpenShift web-console here: https://console-openshift-
console.apps.mycluster.example.com
INFO Login to the console with user: "kubeadmin", and password: "4vYBz-Ee6gm-ymBZj-
Wt5AL"
INFO Time elapsed: 36m22s
```

注意

当安装成功时，集群访问和凭证信息还会输出到 **<installation_directory>/.openshift_install.log**。



重要

- 安装程序生成的 Ignition 配置文件包含在 24 小时后过期的证书，然后在过期时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外情况是，您需要手动批准待处理的 **node-bootstrapper** 证书签名请求（CSR）来恢复 kubelet 证书。如需更多信息，请参阅*从过期的 control plane 证书中恢复*的文档。
- 建议您在生成 12 小时后使用 Ignition 配置文件，因为集群安装后 24 小时证书从 16 小时轮转至 22 小时。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中运行证书更新时避免安装失败。

重要

您不得删除安装程序或安装程序所创建的文件。需要这两者才能删除集群。

3.7.9. 通过下载二进制文件安装 OpenShift CLI

您需要安装 CLI（**oc**） 来使用命令行界面与 OpenShift Container Platform 进行交互。您可在 Linux、Windows 或 macOS 上安装 **oc**。

重要

如果安装了旧版本的 **oc**，则无法使用 OpenShift Container Platform 4.6 中的所有命令。下载并安装新版本的 **oc**。

3.7.9.1. 在 Linux 上安装 OpenShift CLI

您可以按照以下流程在 Linux 上安装 OpenShift CLI（**oc**）二进制文件。

流程

1. 进入到红帽客户门户网站上的 OpenShift Container Platform 下载页面。

2. 在 **Version** 下拉菜单中选择相应的版本。
3. 单击 **OpenShift v4.6 Linux 客户端** 条目旁边的 **Download Now**，再保存文件。
4. 解包存档：

```
$ tar xvzf <file>
```

5. 把 **oc** 二进制代码放到 **PATH** 中的目录中。
   执行以下命令可以查看当前的 **PATH** 设置：

```
$ echo $PATH
```

安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
$ oc <command>
```

3.7.9.2. 在 Windows 上安装 OpenShift CLI

您可以按照以下流程在 Windows 上安装 OpenShift CLI（**oc**）二进制代码。

流程

1. 进入到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 在 **Version** 下拉菜单中选择相应的版本。
3. 单击 **OpenShift v4.6 Windows 客户端**条目旁边的 **Download Now**，再保存文件。
4. 使用 ZIP 程序解压存档。
5. 把 **oc** 二进制代码放到 **PATH** 中的目录中。
   要查看您的 **PATH**，请打开命令提示窗口并执行以下命令：

```
C:\> path
```

安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
C:\> oc <command>
```

3.7.9.3. 在 macOS 上安装 OpenShift CLI

您可以按照以下流程在 macOS 上安装 OpenShift CLI（**oc**）二进制代码。

流程

1. 进入到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 在 **Version** 下拉菜单中选择相应的版本。
3. 单击 **OpenShift v4.6 MacOSX 客户端**条目旁边的 **Download Now**，再保存文件。
4. 解包和解压存档。

5. 将 **oc** 二进制文件移到 PATH 的目录中。
   要查看您的 **PATH**，打开一个终端窗口并执行以下命令：

```
$ echo $PATH
```

安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
$ oc <command>
```

3.7.10. 使用 CLI 登录到集群

您可以通过导出集群 **kubeconfig** 文件，以默认系统用户身份登录集群。**kubeconfig** 文件包含关于集群的信息，供 CLI 用于将客户端连接到正确集群和 API 服务器。该文件特只适用于一个特定的集群，在 OpenShift Container Platform 安装过程中创建。

先决条件

- 已部署了 OpenShift Container Platform 集群。
- 已安装 **oc** CLI。

流程

1. 导出 **kubeadmin** 凭证：

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

   1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

2. 使用导出的配置，验证能否成功运行 **oc** 命令：

```
$ oc whoami
```

输出示例

```
system:admin
```

其他资源

- 如需有关访问和了解 OpenShift Container Platform Web 控制台的更多信息，请参阅访问 Web 控制台。

3.7.11. OpenShift Container Platform 的 Telemetry 访问

在 OpenShift Container Platform 4.6 中，默认运行的 Telemetry 服务提供有关集群健康状况和成功更新的指标，需要访问互联网。如果您的集群连接到互联网，Telemetry 会自动运行，而且集群会注册到 OpenShift Cluster Manager。

确认 OpenShift Cluster Manager 清单正确后，可以由 Telemetry 自动维护，也可以使用 OpenShift Cluster Manager 手动维护，使用订阅监控来跟踪帐户或多集群级别的 OpenShift Container Platform 订阅。

其他资源

- 有关 Telemetry 服务的更多信息，请参阅关于远程健康监控。

3.7.12. 后续步骤

- 自定义集群。
- 如果需要，您可以选择不使用远程健康报告。

3.8. 在 AZURE 上将集群安装到一个政府区域

在 OpenShift Container Platform 版本 4.6 中，您可以在 Microsoft Azure 上将集群安装到一个政府区域。要配置政府区域，请在安装集群前修改 **install-config.yaml** 文件中的参数。

3.8.1. 先决条件

- 查看有关 OpenShift Container Platform 安装和更新流程的详细信息。
- 配置 Azure 帐户 以托管集群，并决定要将集群部署到的已测试和验证的政府区域。
- 如果使用防火墙，则必须将其配置为允许集群需要访问的站点。
- 如果不允许系统管理身份和访问管理（IAM），集群管理员可以 手动创建和维护 IAM 凭证。手动模式也可以用于云 IAM API 无法访问的环境中。

3.8.2. Azure 政府区域

OpenShift Container Platform 支持将集群部署到 Microsoft Azure Government(MAG) 区域。MAG 是为需要运行敏感负载的美国政府机构、企业、企业和其他美国客户特别设计的。MAG 由只有政府数据中心区域组成，它们都赋予了 影响等级 5 授权。

要安装到 MAG 区域，需要在 **install-config.yaml** 文件中手动配置 Azure Government 专用云实例和地区。您还必须更新服务主体以引用适当的政府环境。

注意

Azure 政府区域不能使用安装程序的引导终端提示来选择。您必须在 **install-config.yaml** 文件中手动定义区域。记得还要根据指定的区域设置专用云实例，如 **AzureUSrianCloud**。

3.8.3. 私有集群

您可以部署不公开外部端点的私有 OpenShift Container Platform 集群。私有集群只能从内部网络访问，且无法在互联网中看到。

默认情况下，OpenShift Container Platform 被置备为使用可公开访问的 DNS 和端点。私有集群在部署集群时将 DNS 、Ingress Controller 和 API 服务器设置为私有。这意味着，集群资源只能从您的内部网络访问，且不能在互联网中看到。

要部署私有集群，您必须使用符合您的要求的现有网络。您的集群资源可能会在网络中的其他集群间共享。

另外，您必须从可访问您置备的云的 API 服务、您置备的网络上的主机以及可以连接到互联网来获取安装介质的机器上部署私有集群。您可以使用符合这些访问要求的机器，并按照您的公司规定进行操作。例如，该机器可以是云网络中的堡垒主机，也可以是可通过 VPN 访问网络的机器。

3.8.3.1. Azure 中的私有集群

要在 Microsoft Azure 平台上创建私有集群，您必须提供一个现有的私有 VNet 和子网来托管集群。安装程序还必须能够解析集群所需的 DNS 记录。安装程序只为内部流量配置 Ingress Operator 和 API 服务器。

根据您的网络如何连接到私有 VNET，您可能需要使用 DNS 转发程序来解析集群的专用 DNS 记录。集群的机器在内部使用 **168.63.129.16** 进行 DNS 解析。如需更多信息，请参阅Azure 文档中的 Azure Private DNS？ 和 什么是 IP 地址 168.63.129.16？ 部分 。

集群仍然需要访问互联网来访问 Azure API。

安装私有集群时不需要或创建以下项目：

- **BaseDomainResourceGroup**，因为集群不创建公共记录
- 公共 IP 地址
- 公共 DNS 记录
- 公共端点

> The cluster is configured so that the Operators do not create public records for the cluster and all cluster machines are placed in the private subnets that you specify.

3.8.3.1.1. 限制：

Azure 上的私有集群只受到与使用一个现有 VNet 相关的限制。

3.8.3.2. 用户定义的出站路由

在 OpenShift Container Platform 中，您可以选择自己的出站路由来连接到互联网。这可让您跳过创建公共 IP 地址和公共负载均衡器的步骤。

您可在安装集群前修改 **install-config.yaml** 文件中的参数来配置用户定义的路由。安装集群时，需要一个已存在的 VNet 来使用出站路由，安装程序不负责配置它。

当将集群配置为使用用户定义的路由时，安装程序不会创建以下资源：

- 用于访问互联网的出站规则。
- 公共负载均衡器的公共 IP。
- Kubernetes Service 对象，为出站请求将集群机器添加到公共负载均衡器中。

在设置用户定义的路由前，您必须确保以下项目可用：

- 出口到互联网可以拉取容器镜像，除非使用内部 registry 镜像。
- 集群可以访问 Azure API。
- 配置了各种允许列表端点。您可以在*配置防火墙*部分引用这些端点。

支持一些已存在的网络设置，使用用户定义的路由访问互联网。

带有网络地址转换的专用集群
您可以使用 Azure VNET 网络地址转换(NAT) 为集群中的子网提供出站互联网访问。请参阅 Azure 文档中的使用 Azure CLI 创建 NAT 网关部分。

当使用配置了 Azure NAT 和用户定义的路由的 VNet 设置时，您可以创建没有公共端点的私有集群。

使用 Azure 防火墙的私有集群
您可以使用 Azure Firewall 为用于安装集群的 VNet 提供出站路由。请参阅 Azure 文档中的Azure Firewall 提供用户定义的路由的信息。

使用 Azure Firewall 和用户自定义路由的 VNet 设置时，您可以创建没有公共端点的私有集群。

带有代理配置的私有集群
您可以使用带有用户定义的路由的代理来允许到互联网的出口。您必须确保集群 Operator 不使用代理访问 Azure API。Operator 必须有权访问代理外的 Azure API。

当使用子网的默认路由表时，Azure 会自动填充 **0.0.0.0/0**，所有 Azure API 请求也会通过 Azure 的内部网络路由，即使 IP 地址是公共的。只要网络安全组规则允许出口到 Azure API 端点，您就可以在没有公共端点的情况下创建用户自定义路由的代理。

没有互联网访问的私有集群
您可以安装专用网络，以限制对互联网的所有访问，但 Azure API 除外。这可以通过在本地镜像发行版本镜像 registry 实现。集群必须有权访问以下内容：

- 可以从中拉取容器镜像的内部 registry 镜像
- 访问 Azure API

在满足这些要求时，就可以使用用户定义的路由来创建没有公共端点的私有集群。

3.8.4. 关于为 OpenShift Container Platform 集群重复使用 VNet

在 OpenShift Container Platform 4.6 中，您可以在 Microsoft Azure 中将集群部署到现有的 Azure Virtual Network（VNet）中。如果您这样做，还必须在 VNet 和路由规则中使用现有子网。

通过将 OpenShift Container Platform 部署到现有的 Azure VNet 中，您可能会避开新帐户中的服务限制，或者更容易地利用公司所设置的操作限制。如果您无法获得创建 VNet 所需的基础架构创建权限，则可以使用这个选项。

3.8.4.1. 使用 VNet 的要求

当使用现有 VNet 部署集群时，必须在安装集群前执行额外网络配置。在安装程序置备的基础架构集群中，安装程序通常会创建以下组件，但在安装到现有 VNet 时不会创建它们：

- 子网
- 路由表
- VNets
- 网络安全组

注意

安装程序要求您使用由云提供的 DNS 服务器。不支持使用自定义 DNS 服务器，并导致安装失败。

如果您使用自定义 VNet，您必须正确配置它及其子网，以便安装程序和集群使用。安装程序不能为集群分配要使用的网络范围，为子网设置路由表，或者设置类似 DHCP 的 VNet 选项，因此您必须在安装集群前配置它们。

集群必须能够访问包含现有 VNet 和子网的资源组。虽然集群创建的所有资源都放在它创建的单独资源组中，但有些网络资源则从另外一个独立的组中使用。一些集群 Operator 必须能够访问这两个资源组中的资源。例如，Machine API 控制器会为它创建的虚拟机附加 NICS，使其从网络资源组中划分子网。

您的 VNet 必须满足以下特征：

- VNet 的 CIDR 块必须包含 **Networking.machineCIDR**，它是集群机器的 IP 地址池。
- VNet 及其子网必须属于同一资源组，子网必须配置为使用 Azure 分配的 DHCP IP 地址而不是静态 IP 地址。

您必须在 VNet 中提供两个子网，一个用于 control plane 机器，一个用于计算机器。因为 Azure 在您指定的区域内的不同可用区中分发机器，所以集群将默认具有高可用性功能。

要确保您提供的子网适合您的环境，安装程序会确认以下信息：

- 所有指定的子网都存在。
- 有两个专用子网，一个用于 control plane 机器，一个用于计算机器。
- 子网 CIDR 属于您指定的机器 CIDR。机器不会在没有为其提供私有子网的可用区中置备。如果需要，安装程序会创建管理 control plane 和 worker 节点的公共负载均衡器，Azure 会为其分配一个公共 IP 地址。

注意

如果您销毁了使用现有 VNet 的集群，则不会删除 VNet。

3.8.4.1.1. 网络安全组要求

托管 compute 和 control plane 机器的子网的网络安全组需要特定的访问权限，以确保集群通信正确。您必须创建规则来允许访问所需的集群通信端口。

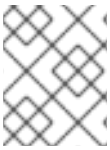
重要

在安装集群前必须先设置网络安全组规则。如果您试图在没有所需访问权限的情况下安装集群，安装程序就无法访问 Azure API，且会导致安装失败。

表 3.24. 所需端口

端口	描述	Control plane	Compute
80	允许 HTTP 流量		x
443	允许 HTTPS 流量		x

端口	描述	Control plane	Compute
6443	允许与 control plane 机器通信。	x	
22623	允许与机器配置服务器通信。	x	



注意

由于集群组件不会修改用户提供的网络安全组（Kubernetes 控制器更新它），因此会创建一个伪网络安全组，供 Kubernetes 控制器修改，而不影响其余的环境。

3.8.4.2. 权限划分

从 OpenShift Container Platform 4.3 开始，您不需要安装程序置备的基础架构集群部署所需的所有权限。这与您所在机构可能已有的权限划分类似：不同的个人可以在您的云中创建不同的资源。例如，您可以创建针对于特定应用程序的对象，如实例、存储和负载均衡器，但不能创建与网络相关的组件，如 VNets 、子网或入站规则。

您在创建集群时使用的 Azure 凭证不需要 VNets 和核心网络组件（如子网、路由表、互联网网关、NAT 和 VPN）所需的网络权限。您仍然需要获取集群中的机器需要的应用程序资源的权限，如负载均衡器 、安全组、存储帐户和节点。

3.8.4.3. 集群间隔离

因为集群无法修改现有子网中的网络安全组，所以无法在 VNet 中相互隔离集群。

3.8.5. OpenShift Container Platform 的互联网访问

在 OpenShift Container Platform 4.6 中，您需要访问互联网来安装集群。

您必须具有以下互联网访问权限：

- 访问 OpenShift Cluster Manager 以下载安装程序并执行订阅管理。如果集群可以访问互联网，并且没有禁用 Telemetry，该服务会自动授权您的集群。
- 访问 Quay.io，以获取安装集群所需的软件包。
- 获取执行集群更新所需的软件包。

重要

如果您的集群无法直接访问互联网，则可以在置备的某些类基础架构上执行受限网络安装。在此过程中，您要下载所需的内容，并使用它在镜像 registry（mirror registry） 中填充安装集群并生成安装程序所需的软件包。对于某些安装类型，集群要安装到的环境不需要访问互联网。在更新集群之前，要更新 registry 镜像系统中的内容。

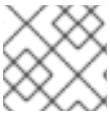
3.8.6. 生成 SSH 私钥并将其添加到代理中

如果要在集群上执行安装调试或灾难恢复，则必须为 **ssh-agent** 和安装程序提供 SSH 密钥。您可以使用此密钥访问公共集群中的 bootstrap 机器来排除安装问题。

注意

在生产环境中，您需要进行灾难恢复和调试。

您可以使用此密钥以 **core** 用户身份通过 SSH 连接到 master 节点。在部署集群时，此密钥会添加到 **core** 用户的 **~/.ssh/authorized_keys** 列表中。



注意

您必须使用一个本地密钥，而不要使用在特定平台上配置的密钥，如 AWS 密钥对。

流程

1. 如果还没有为计算机上免密码身份验证而配置的 SSH 密钥，请创建一个。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ ssh-keygen -t ed25519 -N '' \
  -f <path>/<file_name> 1
```

1 指定新 SSH 密钥的路径和文件名，如 **~/.ssh/id_rsa**。如果您已有密钥对，请确保您的公钥位于 **~/.ssh** 目录中。

运行此命令会在指定的位置生成不需要密码的 SSH 密钥。

注意

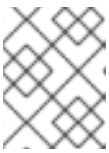
如果您计划在 **x86_64** 架构中安装使用 FIPS 验证的/Modules in Process 加密库的 OpenShift Container Platform 集群，不要创建使用 **ed25519** 算法的密钥。反之，创建一个使用 **rsa** 或 **ecdsa** 算法的密钥。

2. 作为后台任务启动 **ssh-agent** 进程：

```
$ eval "$(ssh-agent -s)"
```

输出示例

```
Agent pid 31874
```



注意

如果您的集群采用 FIPS 模式，则只使用 FIPS 兼容算法来生成 SSH 密钥。密钥必须是 RSA 或 ECDSA。

3. 将 SSH 私钥添加到 **ssh-agent**：

```
$ ssh-add <path>/<file_name> 1
```

输出示例

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

1 指定 SSH 私钥的路径和文件名，如 **~/.ssh/id_rsa**

后续步骤

- 在安装 OpenShift Container Platform 时，为安装程序提供 SSH 公钥。

3.8.7. 获取安装程序

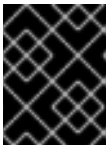
在安装 OpenShift Container Platform 之前，将安装文件下载到本地计算机上。

先决条件

- 运行 Linux 或 macOS 的计算机，本地磁盘空间为 500 MB

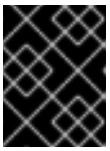
流程

1. 访问 OpenShift Cluster Manager 站点的 Infrastructure Provider 页面。如果您有红帽帐号，请使用自己的凭证登录。如果没有，请创建一个帐户。
2. 选择您的基础架构供应商。
3. 进入适用于您的安装类型的页面，下载您的操作系统的安装程序，并将文件放在要保存安装配置文件的目录中。。



重要

安装程序会在用来安装集群的计算机上创建若干文件。在完成集群安装后，您必须保留安装程序和安装程序所创建的文件。这两个文件都需要删除集群。



重要

删除安装程序创建的文件不会删除您的集群，即使集群在安装过程中失败也是如此。要删除集群，为特定云供应商完成 OpenShift Container Platform 卸载流程。

4. 提取安装程序。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ tar xvf openshift-install-linux.tar.gz
```

5. 从 Red Hat OpenShift Cluster Manager 下载安装 pull secret 。通过此 pull secret，您可以进行所含授权机构提供的服务的身份验证，这些服务包括为 OpenShift Container Platform 组件提供容器镜像的 Quay.io。

3.8.8. 手动创建安装配置文件

在 Microsoft Azure 上安装 OpenShift Container Platform 时，您必须手动生成安装配置文件。

先决条件

- 获取 OpenShift Container Platform 安装程序和集群的访问令牌。

流程

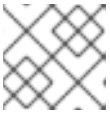
1. 创建用来存储您所需的安装资产的安装目录：

```
$ mkdir <installation_directory>
```

重要

您必须创建目录。一些安装信息，如 bootstrap X.509 证书，有较短的过期间隔，因此不要重复使用安装目录。如果要重复使用另一个集群安装中的个别文件，可以将其复制到您的目录中。但是，一些安装数据的文件名可能会在发行版本之间有所改变。从 OpenShift Container Platform 老版本中复制安装文件时要格外小心。

2. 自定义以下 **install-config.yaml** 文件模板，并将它保存到 **<installation_directory>** 中。



注意

此配置文件必须命名为 **install-config.yaml**。

3. 备份 **install-config.yaml** 文件，以便用于安装多个集群。

重要

install-config.yaml 文件会在安装过程的下一步骤中消耗掉。现在必须备份它。

3.8.8.1. 安装配置参数

在部署 OpenShift Container Platform 集群前，您可以提供参数值，以描述托管集群的云平台的帐户并选择性地自定义集群平台。在创建 **install-config.yaml** 安装配置文件时，您可以通过命令行来提供所需的参数的值。如果要自定义集群，可以修改 **install-config.yaml** 文件来提供关于平台的更多信息。

注意

安装之后，您无法修改 **install-config.yaml** 文件中的这些参数。

重要

openshift-install 命令不验证参数的字段名称。如果指定了不正确的名称，则不会创建相关的文件或对象，且不会报告错误。确保所有指定的参数的字段名称都正确。

3.8.8.1.1. 所需的配置参数

下表描述了所需的安装配置参数：

表 3.25. 所需的参数

参数	描述	值
apiVersion	**install-config.yaml** 内容的 API 版本。当前版本是 **v1**。安装程序还可能支持旧的 API 版本。	字符串

参数	描述	值
baseDomain	云供应商的基域。此基础域用于创建到 OpenShift Container Platform 集群组件的路由。集群的完整 DNS 名称是 **baseDomain** 和 **metadata.name** 参数值的组合，其格式为 **<metadata.name>.<baseDomain>**。	完全限定域名或子域名，如 **example.com**。
metadata	Kubernetes 资源 **ObjectMeta**，其中只消耗 **name** 参数。	对象
metadata.name	集群的名称。集群的 DNS 记录是 **{{.metadata.name}}.{{.baseDomain}}** 的子域。	小写字母,连字符(-)和句点(.)的字符串，如 **dev**。
platform	执行安装的具体平台配置：**aws**、**baremetal**、**azure**、**openstack**、**ovirt**、**vsphere**。有关 **platform.<platform>** 参数的额外信息，请参考下表来了解您的具体平台。	对象

pullSecret	从 Red Hat OpenShift Cluster Manager 获取 pull secret，验证从 Quay.io 等服务中下载 OpenShift Container Platform 组件的容器镜像。	{
		"auths":{
		"cloud.openshift.com":{
		"auth":"b3Blb=",
		"email":"you@example.com"
		},
		"quay.io":{
		"auth":"b3Blb=",
		"email":"you@example.com"
		}
		}
		}

3.8.8.1.2. 网络配置参数

您可以根据现有网络基础架构的要求自定义安装配置。例如，您可以扩展集群网络的 IP 地址块，或者提供不同于默认值的不同 IP 地址块。

只支持 IPv4 地址。

表 3.26. 网络参数

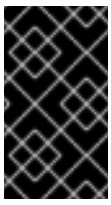
参数	描述	值
networking	集群网络的配置。	对象  **注意** 您不能在安装后修改 **networking** 对象指定的参数。
networking.networkType	要安装的集群网络供应商 Container Network Interface（CNI）插件。	**OpenShiftSDN** 或 **OVNKubernetes**。默认值为 **OpenShiftSDN**。
networking.clusterNetwork	pod 的 IP 地址块。 默认值为 **10.128.0.0/14**，主机前缀为 **/23**。 如果您指定多个 IP 地址块，则块不得互相重叠。	一个对象数组。例如： networking: clusterNetwork: - cidr: 10.128.0.0/14 hostPrefix: 23
networking.clusterNetwork.cidr	使用 **networking.clusterNetwork** 时需要此项。IP 地址块。 一个 IPv4 网络。	使用 CIDR 形式的 IP 地址块。IPv4 块的前缀长度介于 **0** 到 **32** 之间。
networking.clusterNetwork.hostPrefix	分配给每个单独节点的子网前缀长度。例如，如果 **hostPrefix** 设为 **23**，则每个节点从所给的 **cidr** 中分配一个 **/23** 子网。**hostPrefix** 值 **23** 提供 510（2^(32 - 23)- 2）个 pod IP 地址。	子网前缀。 默认值为 **23**。
networking.serviceNetwork	服务的 IP 地址块。默认值为 **172.30.0.0/16**。 OpenShift SDN 和 OVN-Kubernetes 网络供应商只支持服务网络的一个 IP 地址块。	CIDR 格式具有 IP 地址块的数组。例如： networking: serviceNetwork: - 172.30.0.0/16
networking.machineNetwork	机器的 IP 地址块。 如果您指定多个 IP 地址块，则块不得互相重叠。	一个对象数组。例如： networking: machineNetwork: - cidr: 10.0.0.0/16

参数	描述	值
networking.machineNetwork.cidr	使用 **networking.machineNetwork** 时需要。IP 地址块。libvirt 以外的所有平台的默认值为 **10.0.0.0/16**。对于 libvirt，默认值为 **192.168.126.0/24**。	CIDR 表示法中的 IP 网络块。 例如： **10.0.0.0/16**。  **注意** 将 **networking.machineNetwork** 设置为与首选 NIC 所在的 CIDR 匹配。

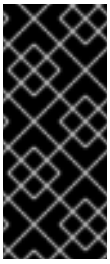
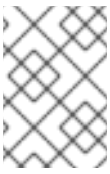
3.8.8.1.3. 可选配置参数

下表描述了可选安装配置参数：

表 3.27. 可选参数

参数	描述	值
additionalTrustBundle	添加到节点可信证书存储中的 PEM 编码 X.509 证书捆绑包。配置了代理时，也可以使用这个信任捆绑包。	字符串
compute	组成计算节点的机器的配置。	machine-pool 对象的数组。详情请查看以下"Machine-pool"表。
compute.architecture	决定池中机器的指令集合架构。目前不支持异构集群，因此所有池都必须指定相同的架构。有效值为 **amd64** （默认值）。	字符串
compute.hyperthreading	是否在计算机器上启用或禁用并发多线程或**超线程**。默认情况下，启用并发多线程以提高机器内核的性能。  **重要** 如果禁用并发多线程，请确保在容量规划时考虑到机器性能可能会显著降低的问题。	**Enabled** 或 **Disabled**
compute.name	使用 **compute** 时需要此值。机器池的名称。	**worker**

参数	描述	值
compute.platform	使用 **compute** 时需要此值。使用此参数指定托管 worker 机器的云供应商。此参数值必须与 **controlPlane.platform** 参数值匹配。	**aws**、**azure**、**gcp**、**openstack**、**ovirt**、**vsphere** 或 **{}**
compute.replicas	要置备的计算机器数量，也称为 worker 机器。	大于或等于 **2** 的正整数。默认值为 **3**。
controlPlane	组成 control plane 的机器的配置。	**MachinePool** 对象的数组。详情请查看以下"Machine-pool"表。
controlPlane.architecture	决定池中机器的指令集合架构。目前不支持异构集群，因此所有池都必须指定相同的架构。有效值为 **amd64** （默认值）。	字符串
controlPlane.hyperthreading	是否在 control plane 机器上启用或禁用并发多线程或**超线程**。默认情况下，启用并发多线程以提高机器内核的性能。  **重要** 如果禁用并发多线程，请确保在容量规划时考虑到机器性能可能会显著降低的问题。	**Enabled** 或 **Disabled**
controlPlane.name	使用 **controlPlane** 时需要。机器池的名称。	**master**
controlPlane.platform	使用 **controlPlane** 时需要。使用此参数指定托管 control plane 机器的云供应商。此参数值必须与 **compute.platform** 参数值匹配。	**aws**、**azure**、**gcp**、**openstack**、**ovirt**、**vsphere** 或 **{}**
controlPlane.replicas	要置备的 control plane 机器数量。	唯一支持的值是 **3**，它是默认值。

参数	描述	值
credentialsMode	Cloud Credential Operator（CCO）模式。如果没有指定任何模式，CCO 会动态地尝试决定提供的凭证的功能，在支持多个模式的平台上使用 mint 模式。  **注意** 不是所有 CCO 模式都支持所有云供应商。如需有关 CCO 模式的更多信息，请参阅*Red Hat Operator 参考指南*内容中的 *Cloud Credential Operator* 条目。	**Mint**、**Passthrough**、**Manual** 或空字符串(**""**)。
fips	启用或禁用 FIPS 模式。默认为 **false**（禁用）。如果启用了 FIPS 模式，运行 OpenShift Container Platform 的 Red Hat Enterprise Linux CoreOS (RHCOS) 机器会绕过默认的 Kubernetes 加密套件，并使用由 RHCOS 提供的加密模块。  **重要** 只有在 **x86_64** 架构中的 OpenShift Container Platform 部署支持 FIPS 验证的/Modules in Process 加密库。  **注意** 如果使用 Azure File 存储，则无法启用 FIPS 模式。	**false** 或 **true**
imageContentSources	release-image 内容的源和仓库。	对象数组。包括一个 **source** 以及可选的 **mirrors**，如下表所示。
imageContentSources.source	使用 **imageContentSources** 时需要。指定用户在镜像拉取规格中引用的仓库。	字符串
imageContentSources.mirrors	指定可能还包含同一镜像的一个或多个仓库。	字符串数组

参数	描述	值
publish	如何发布或公开集群的面向用户的端点，如 Kubernetes API、OpenShift 路由。	**Internal** 或 **External**。把 **publish** 设置为 **Internal** 以部署一个私有集群，它不能被互联网访问。默认值为 **External**。

sshKey	用于验证集群机器访问的 SSH 密钥或密钥。
	 
	注意
	对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 **ssh-agent** 进程使用的 SSH 密钥。
		一个或多个密钥。例如：
		`sshKey:`
		`  <key1>`
		`  <key2>`
		`  <key3>`

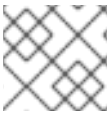
3.8.8.1.4. 其他 Azure 配置参数

下表描述了其他 Azure 配置参数：

表 3.28. 其他 Azure 参数

参数	描述	值
compute.platform.azure.osDisk.diskSizeGB	虚拟机的 Azure 磁盘大小。	以 GB 为单位表示磁盘大小的整数。默认值为 **128**。
compute.platform.azure.osDisk.diskType	定义磁盘的类型。	**Standard_LRS**、**Premium_LRS** 或 **标准SSD_LRS**。默认值为 **Premium_LRS**。
controlPlane.platform.azure.osDisk.diskSizeGB	虚拟机的 Azure 磁盘大小。	以 GB 为单位表示磁盘大小的整数。默认值为 **1024**。
controlPlane.platform.azure.osDisk.diskType	定义磁盘的类型。	**Premium_LRS** 或 **标准SSD_LRS**. 默认值为 **Premium_LRS**。
platform.azure.baseDomainResourceGroupName	包含基域的 DNS 区的资源组的名称。	字符串，如 **production_cluster**。

参数	描述	值
platform.azure.outboundType	用于将集群连接到互联网的出站路由策略。如果使用用户定义的路由，则必须在安装集群前配置出站路由。安装程序不负责配置用户定义的路由。	**loadbalancer** 或 **UserDefinedRouting**。默认为 **LoadBalancer**。
platform.azure.region	托管集群的 Azure 区域名称。	任何有效的区域名称，如 **centralus**。
platform.azure.zone	可在其中放入机器的可用区的列表。如需高可用性，请至少指定两个区域。	区域列表，如 **["1", "2", "3"]**。
platform.azure.networkResourceGroupName	包含要将集群部署到的现有 VNet 的资源组名称。这个名称不能和 **platform.azure.baseDomainResourceGroupName**相同。	字符串。
platform.azure.virtualNetwork	要将集群部署到的现有 VNet 的名称。	字符串。
platform.azure.controlPlaneSubnet	要将 control plane 机器部署到的 VNet 中现有子网的名称。	有效的 CIDR，如 **10.0.0.0/16**。
platform.azure.computeSubnet	您要将计算机器部署到的 VNet 中现有子网的名称。	有效的 CIDR，如 **10.0.0.0/16**。
platform.azure.cloudName	用于使用适当 Azure API 端点配置 Azure SDK 的 Azure 云环境名称。如果为空，则使用默认值 **AzurePublicCloud**。	任何有效的云环境，如 **AzurePublicCloud** 或 **AzureUSGovernmentCloud**。



注意

您无法自定义 Azure 可用区，也不能使用标签来整理用于 Azure 集群的 Azure 资源。

3.8.8.2. Azure 的自定义 install-config.yaml 文件示例

您可以自定义 **install-config.yaml** 文件，以指定有关 OpenShift Container Platform 集群平台的更多信息，或修改所需参数的值。

重要

此示例 YAML 文件仅供参考。您必须使用安装程序来获取 **install-config.yaml** 文件，并且修改该文件。

```
apiVersion: v1
baseDomain: example.com (1)
controlPlane: (2)
  hyperthreading: Enabled (3) (4)
  name: master
```

```
    platform:
      azure:
        osDisk:
          diskSizeGB: 1024 5
          diskType: Premium_LRS
        type: Standard_D8s_v3
    replicas: 3
  compute: 6
  - hyperthreading: Enabled 7
    name: worker
    platform:
      azure:
        type: Standard_D2s_v3
        osDisk:
          diskSizeGB: 512 8
          diskType: Standard_LRS
        zones: 9
        - "1"
        - "2"
        - "3"
    replicas: 5
  metadata:
    name: test-cluster 10
  networking:
    clusterNetwork:
    - cidr: 10.128.0.0/14
      hostPrefix: 23
    machineNetwork:
    - cidr: 10.0.0.0/16
    networkType: OpenShiftSDN
    serviceNetwork:
    - 172.30.0.0/16
  platform:
    azure:
      baseDomainResourceGroupName: resource_group 11
      region: usgovvirginia
      resourceGroupName: existing_resource_group 12
      networkResourceGroupName: vnet_resource_group 13
      virtualNetwork: vnet 14
      controlPlaneSubnet: control_plane_subnet 15
      computeSubnet: compute_subnet 16
      outboundType: UserDefinedRouting 17
      cloudName: AzureUSGovernmentCloud 18
  pullSecret: '{"auths": ...}' 19
  fips: false 20
  sshKey: ssh-ed25519 AAAA... 21
  publish: Internal 22
```

1 10 19 必需。

2 6 如果没有提供这些参数和值，安装程序会提供默认值。

3 7 **controlPlane** 部分是一个单映射，但 **compute** 部分是一系列映射。为满足不同数据结构的要求，**compute** 部分的第一行必须以连字符 **-** 开头，**controlPlane** 部分的第一行则不可以连字符开

头。只使用一个 control plane 池。

4 是否要启用或禁用并发多线程或**超线程**。默认情况下，启用并发多线程以提高机器内核的性能。您可以通过将参数值设为 **Disabled** 来禁用。如果您在某些集群机器上禁用并发多线程，则必须在所有集群机器上禁用。

重要

如果禁用并发多线程，请确保在容量规划时考虑到机器性能可能会显著降低的问题。如果禁用并发多线程，请使用较大的虚拟机类型，如 **Standard_D8s_v3**。

5 8 可以 GB 为单位指定要使用的磁盘大小。control plane 节点（也称为 master 节点）的最低推荐值为 1024 GB。

9 指定要将机器部署到的区域列表。如需高可用性，请至少指定两个区域。

11 指定包含基域的 DNS 区的资源组的名称。

12 指定要安装集群的现有资源组的名称。如果未定义，则会为集群创建新的资源组。

13 如果您使用现有的 如果使用现有的 VNet，请指定包含它的资源组的名称。。

14 如果使用现有的 VNet，请指定其名称。

15 如果使用现有的 VNet，请指定托管 control plane 机器的子网名称。

16 如果使用现有的 VNet，请指定托管计算机器的子网名称。

17 您可以自定义自己的出站路由。配置用户定义的路由可防止在集群中公开外部端点。出口的用户自定义路由需要将集群部署到现有的 VNet。

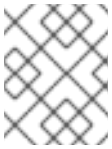
18 指定要将集群部署到的 Azure 云环境的名称。将 **AzureUSüCloud** 设置为部署至 Microsoft Azure Government（MAG）区域。默认值为 **AzurePublicCloud**。

20 是否启用或禁用 FIPS 模式。默认情况下不启用 FIPS 模式。如果启用了 FIPS 模式，运行 OpenShift Container Platform 的 Red Hat Enterprise Linux CoreOS (RHCOS) 机器会绕过默认的 Kubernetes 加密套件，并使用由 RHCOS 提供的加密模块。

重要

只有在 **x86_64** 架构中的 OpenShift Container Platform 部署支持 FIPS 验证的/Modules in Process 加密库。

21 您可以选择提供您用来访问集群中机器的 **sshKey** 值。



注意

对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 **ssh-agent** 进程使用的 SSH 密钥。

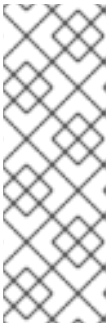
22 如何发布集群的面向用户的端点。把 **publish** 设置为 **Internal** 以部署一个私有集群，它不能被互联网访问。默认值为 **External**。

3.8.8.3. 在安装过程中配置集群范围代理

生产环境可能会拒绝直接访问互联网，而是提供 HTTP 或 HTTPS 代理。您可以通过在 **install-config.yaml** 文件中配置代理设置，将新的 OpenShift Container Platform 集群配置为使用代理。

先决条件

- 您有一个现有的 **install-config.yaml** 文件。
- 您检查了集群需要访问的站点，并决定是否需要绕过代理。默认情况下代理所有集群出口流量，包括对托管云供应商 API 的调用。您需要将站点添加到 **Proxy** 对象的 **spec.noProxy** 字段来绕过代理。



注意

Proxy 对象 **status.noProxy** 字段使用安装配置中的 **networking.machineNetwork[].cidr**、**networking.clusterNetwork[].cidr** 和 **networking.serviceNetwork[]** 字段的值填充。

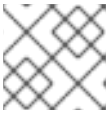
对于在 Amazon Web Services(AWS)、Google Cloud Platform(GCP)、Microsoft Azure 和 Red Hat OpenStack Platform(RHOSP)上安装, **Proxy** 对象 **status.noProxy** 字段也会使用实例元数据端点填充(**169.254.169.254**)。

流程

1. 编辑 **install-config.yaml** 文件并添加代理设置。例如：

```
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: example.com 3
additionalTrustBundle: | 4
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
...
```

1. 用于创建集群外 HTTP 连接的代理 URL。URL 必须是 **http**。
2. 用于创建集群外 HTTPS 连接的代理 URL。
3. 要排除在代理中的目标域名、IP 地址或其他网络 CIDR 的逗号分隔列表。在域前面加 **.** 来仅匹配子域。例如： **.y.com** 匹配 **x.y.com**，但不匹配 **y.com**。使用 ***** 绕过所有目的地的代理。
4. 如果提供，安装程序会在 **openshift-config 命名空间中生成名为 user-ca- bundle** 的配置映射来保存额外的 CA 证书。如果您提供 **additionalTrustBundle** 和至少一个代理设置，则 **Proxy** 对象会被配置为引用 **trustedCA** 字段中的 **user-ca-bundle** 配置映射。然后，Cluster Network Operator 会创建一个 **trusted-ca-bundle** 配置映射，该配置映射将为 **trustedCA** 参数指定的内容与 RHCOS 信任捆绑包合并。 **additionalTrustBundle** 字段是必需的，除非代理的身份证书由来自 RHCOS 信任捆绑包的颁发机构签名。



注意

安装程序不支持代理的 **readinessEndpoints** 字段。

2. 保存该文件，并在安装 OpenShift Container Platform 时引用。

安装程序会创建一个名为 **cluster** 的集群范围代理，该代理使用提供的 **install-config.yaml** 文件中的代理设置。如果没有提供代理设置，仍然会创建一个 **cluster Proxy** 对象，但它会有一个空 **spec**。

注意

只支持名为 **cluster** 的 **Proxy** 对象，且无法创建额外的代理。

3.8.9. 部署集群

您可以在兼容云平台中安装 OpenShift Container Platform。

重要

安装程序的 **create cluster** 命令只能在初始安装过程中运行一次。

先决条件

- 配置托管集群的云平台的帐户。
- 获取 OpenShift Container Platform 安装程序以及集群的 pull secret。

流程

1. 更改为包含安装程序的目录并初始化集群部署：

```
$ ./openshift-install create cluster --dir <installation_directory> \ 1
    --log-level=info 2
```

1 对于 **<installation_directory>**，请指定自定义 **./install-config.yaml** 文件的位置。

2 要查看不同的安装详情，请指定 **warn**、**debug** 或 **error**，而不要指定 **info**。

注意

如果您在主机上配置的云供应商帐户没有足够的权限来部署集群，安装过程将会停止，并且显示缺少的权限。

集群部署完成后，终端会显示访问集群的信息，包括指向其 Web 控制台的链接和 **kubeadmin** 用户的凭证。

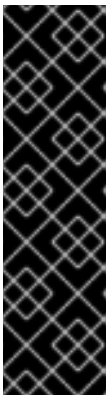
输出示例

```
...
INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run 'export
KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'
```

```
INFO Access the OpenShift web-console here: https://console-openshift-
console.apps.mycluster.example.com
INFO Login to the console with user: "kubeadmin", and password: "4vYBz-Ee6gm-ymBZj-
Wt5AL"
INFO Time elapsed: 36m22s
```

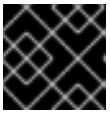
注意

当安装成功时，集群访问和凭证信息还会输出到 **<installation_directory>/.openshift_install.log**。



重要

- 安装程序生成的 Ignition 配置文件包含在 24 小时后过期的证书，然后在过期时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外情况是，您需要手动批准待处理的 **node-bootstrapper** 证书签名请求（CSR）来恢复 kubelet 证书。如需更多信息，请参阅*从过期的 control plane 证书中恢复*的文档。
- 建议您在生成 12 小时后使用 Ignition 配置文件，因为集群安装后 24 小时证书从 16 小时轮转至 22 小时。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中运行证书更新时避免安装失败。

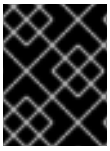


重要

您不得删除安装程序或安装程序所创建的文件。需要这两者才能删除集群。

3.8.10. 通过下载二进制文件安装 OpenShift CLI

您需要安装 CLI（**oc**） 来使用命令行界面与 OpenShift Container Platform 进行交互。您可在 Linux、Windows 或 macOS 上安装 **oc**。



重要

如果安装了旧版本的 **oc**，则无法使用 OpenShift Container Platform 4.6 中的所有命令。下载并安装新版本的 **oc**。

3.8.10.1. 在 Linux 上安装 OpenShift CLI

您可以按照以下流程在 Linux 上安装 OpenShift CLI（**oc**）二进制文件。

流程

1. 进入到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 在 **Version** 下拉菜单中选择相应的版本。
3. 单击 **OpenShift v4.6 Linux 客户端** 条目旁边的 **Download Now**，再保存文件。
4. 解包存档：

```
$ tar xvzf <file>
```

5. 把 **oc** 二进制代码放到 **PATH** 中的目录中。
   执行以下命令可以查看当前的 **PATH** 设置：

```
$ echo $PATH
```

安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
$ oc <command>
```

3.8.10.2. 在 Windows 上安装 OpenShift CLI

您可以按照以下流程在 Windows 上安装 OpenShift CLI（**oc**）二进制代码。

流程

1. 进入到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 在 **Version** 下拉菜单中选择相应的版本。
3. 单击 **OpenShift v4.6 Windows 客户端**条目旁边的 **Download Now**，再保存文件。
4. 使用 ZIP 程序解压存档。
5. 把 **oc** 二进制代码放到 **PATH** 中的目录中。
   要查看您的 **PATH**，请打开命令提示窗口并执行以下命令：

```
C:\> path
```

安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
C:\> oc <command>
```

3.8.10.3. 在 macOS 上安装 OpenShift CLI

您可以按照以下流程在 macOS 上安装 OpenShift CLI（**oc**）二进制代码。

流程

1. 进入到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 在 **Version** 下拉菜单中选择相应的版本。
3. 单击 **OpenShift v4.6 MacOSX 客户端**条目旁边的 **Download Now**，再保存文件。
4. 解包和解压存档。
5. 将 **oc** 二进制文件移到 PATH 的目录中。
   要查看您的 **PATH**，打开一个终端窗口并执行以下命令：

```
$ echo $PATH
```

安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
$ oc <command>
```

3.8.11. 使用 CLI 登录到集群

您可以通过导出集群 **kubeconfig** 文件，以默认系统用户身份登录集群。**kubeconfig** 文件包含关于集群的信息，供 CLI 用于将客户端连接到正确集群和 API 服务器。该文件特只适用于一个特定的集群，在 OpenShift Container Platform 安装过程中创建。

先决条件

- 已部署了 OpenShift Container Platform 集群。
- 已安装 **oc** CLI。

流程

1. 导出 **kubeadmin** 凭证：

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

2. 使用导出的配置，验证能否成功运行 **oc** 命令：

```
$ oc whoami
```

输出示例

```
system:admin
```

其他资源

- 如需有关访问和了解 OpenShift Container Platform Web 控制台的更多信息，请参阅访问 Web 控制台。

3.8.12. OpenShift Container Platform 的 Telemetry 访问

在 OpenShift Container Platform 4.6 中，默认运行的 Telemetry 服务提供有关集群健康状况和成功更新的指标，需要访问互联网。如果您的集群连接到互联网，Telemetry 会自动运行，而且集群会注册到 OpenShift Cluster Manager。

确认 OpenShift Cluster Manager 清单正确后，可以由 Telemetry 自动维护，也可以使用 OpenShift Cluster Manager 手动维护，使用订阅监控来跟踪帐户或多集群级别的 OpenShift Container Platform 订阅。

其他资源

- 有关 Telemetry 服务的更多信息，请参阅关于远程健康监控。

3.8.13. 后续步骤

- 自定义集群。
- 如果需要，您可以选择不使用远程健康报告。

3.9. 详情请参阅在使用 ARM 模板的 AZURE 上安装集群。

在 OpenShift Container Platform 版本 4.6 中，您可以使用您提供的基础架构在 Microsoft Azure 上安装集群。

提供的几个 Azure Resource Manager（ARM） 模板可协助完成这些步骤，也可帮助您自行建模。

重要

进行用户置备的基础架构安装的步骤仅作为示例。使用您提供的基础架构安装集群需要了解云供应商和 OpenShift Container Platform 安装过程。提供的几个 ARM 模板可帮助完成这些步骤，或帮助您自行建模。您也可以自由选择通过其他方法创建所需的资源；模板仅作示例之用。

3.9.1. 先决条件

- 查看有关 OpenShift Container Platform 安装和更新流程的详细信息。
- 配置 Azure 帐户以托管集群。
- 下载 Azure CLI 并安装到您的计算机上。请参阅 Azure 文档中的安装 Azure CLI。以下文档使用 Azure CLI 版本 **2.2.0** 进行测试。Azure CLI 命令可能会根据您使用的版本的不同而不同。
- 如果使用防火墙并计划使用遥测（telemetry），您必须将防火墙配置为允许集群需要访问的站点。
- 如果不允许系统管理身份和访问管理（IAM），集群管理员可以 手动创建和维护 IAM 凭证。手动模式也可以用于云 IAM API 无法访问的环境中。

注意

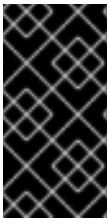
如果您要配置代理，请务必也要查看此站点列表。

3.9.2. OpenShift Container Platform 的互联网访问

在 OpenShift Container Platform 4.6 中，您需要访问互联网来安装集群。

您必须具有以下互联网访问权限：

- 访问 OpenShift Cluster Manager 以下载安装程序并执行订阅管理。如果集群可以访问互联网，并且没有禁用 Telemetry，该服务会自动授权您的集群。
- 访问 Quay.io，以获取安装集群所需的软件包。
- 获取执行集群更新所需的软件包。



重要

如果您的集群无法直接访问互联网，则可以在置备的某些类基础架构上执行受限网络安装。在此过程中，您要下载所需的内容，并使用它在镜像 registry（mirror registry） 中填充安装集群并生成安装程序所需的软件包。对于某些安装类型，集群要安装到的环境不需要访问互联网。在更新集群之前，要更新 registry 镜像系统中的内容。

3.9.3. 配置 Azure 项目

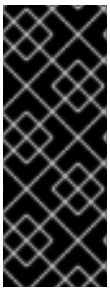
在安装 OpenShift Container Platform 之前，您必须配置 Azure 项目来托管它。

重要

所有通过公共端点提供的 Azure 资源均存在资源名称的限制，您无法创建使用某些名称的资源。如需 Azure 限制词语列表，请参阅 Azure 文档中的解决保留资源名称错误。

3.9.3.1. Azure 帐户限值

OpenShift Container Platform 集群使用诸多 Microsoft Azure 组件，默认的 Azure 订阅和服务限值、配额和约束会影响您安装 OpenShift Container Platform 集群的能力。



重要

默认的限制因服务类别的不同（如 Free Trial 或 Pay-As-You-Go）以及系列的不同（如 Dv2、F 或 G）而有所不同。例如，对于 Enterprise Agreement 订阅的默认限制是 350 个内核。

在 Azure 上安装默认集群前，请检查您的订阅类型的限制，如有必要，请提高帐户的配额限制。

下表总结了 Azure 组件，它们的限值会影响您安装和运行 OpenShift Container Platform 集群的能力。

组件	默认所需的组件数	默认 Azure 限值	描述

组件	默认所需的组件数	默认 Azure 限值	描述
vCPU	40	每个区域 20 个	默认集群需要 40 个 vCPU，因此您必须提高帐户限值。 默认情况下，每个集群创建以下实例： ● 一台 Bootstrap 机器，在安装后删除 ● 三个 control plane 机器 ● 三个计算（compute）机器 由于 Bootstrap 机器使用 **Standard_D4s_v3** 机器（使用 4 个 vCPU），control plane 机器使用 **Standard_D8s_v3** 虚拟机（8 个 vCPU），并且 worker 机器使用 **Standard_D4s_v3** 虚拟机（4 个 vCPU），因此默认集群需要 40 个 vCPU。bootstrap 节点 VM（使用 4 个 vCPU）只在安装过程中使用。 若要部署更多 worker 节点、启用自动扩展、部署大型工作负载或使用不同的实例类型，您必须进一步提高帐户的 vCPU 限值，以确保集群可以部署您需要的机器。 默认情况下，安装程序将 control plane 和 compute 机器分布到一个区域中的所有可用区。要确保集群的高可用性，请选择至少含有三个可用区的区域。如果您的区域包含的可用区少于三个，安装程序将在可用区中放置多台 control plane 机器。
OS Disk	7		虚拟机 OS 磁盘必须能够保持最低 5000 IOPS/200MBps 的吞吐量。此吞吐量可以通过至少 1 TiB Premium SSD (P30) 提供。在 Azure 中，磁盘性能直接依赖于 SSD 磁盘大小，因此要达到 **Standard_D8s_v3** 支持的吞吐量，或其他类似的机器类型，目标为 5000 IOPS，则至少需要一个 P30 磁盘。 主机缓存必须设置为 **ReadOnly** 以获得低读取延迟和高读取 IOPS 和吞吐量。从缓存执行的读取可能存在于虚拟机内存或本地 SSD 磁盘中，比数据磁盘的读取速度要快得多，因为数据磁盘位于 blob 存储中。
VNet	1	每个区域 1000 个	每个默认集群都需要一个虚拟网络 (VNet)，此网络包括两个子网。
网络接口	6	每个区域 65,536 个	每个默认集群都需要六个网络接口。如果您要创建更多机器或者您部署的工作负载要创建负载均衡器，则集群会使用更多的网络接口。

组件	默认所需的组件数	默认 Azure 限值	描述
网络安全组	2	5000	每个默认集群为 VNet 中的每个子网创建网络安全组。默认集群为 control plane 和计算节点子网创建网络安全组： **control plane**：允许从任何位置通过端口 6443 访问 control plane 机器 **node**：允许从互联网通过端口 80 和 443 访问 worker 节点
网络负载均衡器	3	每个区域 1000 个	每个集群都会创建以下负载均衡器： **default**：用于在 worker 机器之间对端口 80 和 443 的请求进行负载均衡的公共 IP 地址 **internal**：用于在 control plane 机器之间对端口 6443 和 22623 的请求进行负载均衡的专用 IP 地址 **external**：用于在 control plane 机器之间对端口 6443 的请求进行负载均衡的公共 IP 地址 如果您的应用程序创建了更多的 Kubernetes **LoadBalancer** 服务对象，您的集群会使用更多的负载均衡器。
公共 IP 地址	3		两个公共负载均衡器各自使用一个公共 IP 地址。bootstrap 机器也使用一个公共 IP 地址，以便您可以在安装期间通过 SSH 连接到该机器来进行故障排除。bootstrap 节点的 IP 地址仅在安装过程中使用。
专用 IP 地址	7		内部负载均衡器、三台 control plane 机器中的每一台以及三台 worker 机器中的每一台各自使用一个专用 IP 地址。
Spot VM vCPU（可选）	0 如果配置 spot 虚拟机，您的集群必须为每个计算节点有两个 spot VM vCPU。	每个区域 20 个	这是可选组件。要使用 spot 虚拟机，您必须将 Azure 默认限值增加到集群中至少有两倍的计算节点数量。 **注意** 不建议将 spot 虚拟机用于 control plane 节点。

3.9.3.2. 在 Azure 中配置公共 DNS 区

要安装 OpenShift Container Platform，您使用的 Microsoft Azure 帐户必须在帐户中具有一个专用的公共托管 DNS 区。此区域必须对域具有权威。此服务为集群外部连接提供集群 DNS 解析和名称查询。

流程

1. 标识您的域或子域，以及注册商（registrar）。您可以转移现有的域和注册商，或通过 Azure 或其他来源获取新的域和注册商。

注意

如需通过 Azure 购买域的更多信息，请参阅 Azure 文档中的购买 Azure 应用服务的自定义域名。

2. 如果您使用现有的域和注册商，请将其 DNS 迁移到 Azure。请参阅 Azure 文档中的将活动 DNS 名称迁移到 Azure 应用服务。

3. 为您的域配置 DNS。按照 Azure 文档中教程：在 Azure DNS 中托管域部分里的步骤，为您的域或子域创建一个公共托管区，提取新的权威名称服务器，并更新您的域使用的名称服务器的注册商记录。
使用合适的根域（如 **openshiftcorp.com**）或子域（如 **clusters.openshiftcorp.com**）。

4. 如果您使用子域，请按照您公司的流程将其委派记录添加到父域。

您可以通过访问此 示例来创建 DNS 区域来查看 Azure 的 DNS 解决方案 。

3.9.3.3. 提高 Azure 帐户限值

要提高帐户限值，请在 Azure 门户上提交支持请求。

注意

每一支持请求只能提高一种类型的配额。

流程

1. 从 Azure 门户，点击左下角的 **Help + suport**。

2. 点击 **New support request**，然后选择所需的值：

   a. 从 **Issue type** 列表中，选择 **Service and subscription limits (quotas)**。

   b. 从 **Subscription** 列表中，选择要修改的订阅。

   c. 从 **Quota type** 列表中，选择要提高的配额。例如，选择 **Compute-VM (cores-vCPUs) subscription limit increases** 以增加 vCPU 的数量，这是安装集群所必须的。

   d. 点击 **Next: Solutions**。

3. 在 **Problem Details** 页面中，提供您要提高配额所需的信息：

   a. 点击 **Provide details**，然后在 **Quota details** 窗口中提供所需的详情。

   b. 在 SUPPORT METHOD 和 CONTACT INFO 部分中，提供问题严重性和您的联系详情。

4. 点击 **Next: Review + create**，然后点击 **Create**。

3.9.3.4. 证书签名请求管理

在使用您置备的基础架构时，集群只能有限地访问自动机器管理，因此您必须提供一种在安装后批准集群证书签名请求 (CSR) 的机制。**kube-controller-manager** 只能批准 kubelet 客户端 CSR。 **machine-approver** 无法保证使用 kubelet 凭证请求的提供证书的有效性，因为它不能确认是正确的机器发出了该请求。您必须决定并实施一种方法，以验证 kubelet 提供证书请求的有效性并进行批准。

3.9.3.5. 所需的 Azure 角色

OpenShift Container Platform 需要一个服务主体，以便可以管理 Microsoft Azure 资源。在创建服务主体前，您的 Azure 帐户订阅必须具有以下角色：

- **User Access Administrator**
- **所有者**

要在 Azure 门户上设置角色，请参阅 Azure 文档中的使用 RBAC 和 Azure 门户管理对 Azure 资源的访问。

3.9.3.6. 创建服务主体

由于 OpenShift Container Platform 及其安装程序必须通过 Azure Resource Manager 创建 Microsoft Azure 资源，因此您必须创建一个能代表它的服务主体。

先决条件

- 安装或更新 Azure CLI。
- 安装**jq**软件包。
- 您的 Azure 帐户具有您所用订阅所需的角色。

流程

1. 登录 Azure CLI：

```
$ az login
```

   在 Web 控制台中，使用您的凭证登录 Azure。

2. 如果您的 Azure 帐户使用订阅，请确保使用正确的订阅。
   a. 查看可用帐户列表并记录您要用于集群的订阅的 **tenantId** 值：

```
$ az account list --refresh
```

   输出示例

```
[
  {
    "cloudName": "AzureCloud",
    "id": "9bab1460-96d5-40b3-a78e-17b15e978a80",
```

```
    "isDefault": true,
    "name": "Subscription Name",
    "state": "Enabled",
    "tenantId": "6057c7e9-b3ae-489d-a54e-de3f6bf6a8ee",
    "user": {
      "name": "you@example.com",
      "type": "user"
    }
  }
]
```

b. 查看您的活跃帐户详情，确认 **tenantId** 值与您要使用的订阅匹配：

```
$ az account show
```

输出示例

```
{
  "environmentName": "AzureCloud",
  "id": "9bab1460-96d5-40b3-a78e-17b15e978a80",
  "isDefault": true,
  "name": "Subscription Name",
  "state": "Enabled",
  "tenantId": "6057c7e9-b3ae-489d-a54e-de3f6bf6a8ee", (1)
  "user": {
    "name": "you@example.com",
    "type": "user"
  }
}
```

(1) 确定 **tenantId** 参数的值是正确订阅的 UUID。

c. 如果您使用的订阅不正确，请更改活跃的订阅：

```
$ az account set -s <id> (1)
```

(1) 替换您要用于 **<id>** 的订阅的 **id** 值。

d. 如果您更改了活跃订阅，请重新显示您的帐户信息：

```
$ az account show
```

输出示例

```
{
  "environmentName": "AzureCloud",
  "id": "33212d16-bdf6-45cb-b038-f6565b61edda",
  "isDefault": true,
  "name": "Subscription Name",
  "state": "Enabled",
  "tenantId": "8049c7e9-c3de-762d-a54e-dc3f6be6a7ee",
  "user": {
```

```
    "name": "you@example.com",
    "type": "user"
  }
}
```

3. 记录前面输出中 **tenantId** 和 **id** 参数的值。OpenShift Container Platform 安装过程中需要这些值。

4. 为您的帐户创建服务主体：

```
$ az ad sp create-for-rbac --role Contributor --name <service_principal> 1
```

1 将 **<service_principal>** 替换要您要分配给服务主体的名称。

输出示例

```
Changing "<service_principal>" to a valid URI of "http://<service_principal>", which is the
required format used for service principal names
Retrying role assignment creation: 1/36
Retrying role assignment creation: 2/36
Retrying role assignment creation: 3/36
Retrying role assignment creation: 4/36
{
  "appId": "8bd0d04d-0ac2-43a8-928d-705c598c6956",
  "displayName": "<service_principal>",
  "name": "http://<service_principal>",
  "password": "ac461d78-bf4b-4387-ad16-7e32e328aec6",
  "tenant": "6048c7e9-b2ad-488d-a54e-dc3f6be6a7ee"
}
```

5. 记录前面输出中 **appId** 和 **password** 参数的值。OpenShift Container Platform 安装过程中需要这些值。

6. 为服务主体授予额外权限。

   - 您必须始终将 **Contributor** 和 **User Access Administrator** 角色添加到应用程序注册服务主体中，以便集群可以为组件分配凭证。
   - 要以 *mint 模式*操作 Cloud Credential Operator（CCO），应用程序注册服务主体还需要 **Azure Active Directory Graph/Application.ReadWrite.OwnedBy** API 权限。
   - 要以 *passthrough 模式* 操作 CCO，应用程序注册服务主体不需要额外的 API 权限。

   如需有关 CCO 模式的更多信息，请参阅 **Red Hat Operator** 参考内容中的 **Cloud Credential Operator** 条目。

   a. 要分配 **User Access Administrator** 角色，请运行以下命令：

```
$ az role assignment create --role "User Access Administrator" \
    --assignee-object-id $(az ad sp list --filter "appId eq '<appId>'" \
      | jq '.[0].id' -r) 1
```

1 将 **<appId>** 替换为服务器主体的 **appId** 参数值。

b. 要分配 **Azure Active Directory Graph** 权限，请运行以下命令：

```
$ az ad app permission add --id <appId> \ 1
    --api 00000002-0000-0000-c000-000000000000 \
    --api-permissions 824c81eb-e3f8-4ee6-8f6d-de7f50d565b7=Role
```

1 将 **<appId>** 替换为服务器主体的 **appId** 参数值。

输出示例

```
Invoking "az ad app permission grant --id 46d33abc-b8a3-46d8-8c84-f0fd58177435 --api
00000002-0000-0000-c000-000000000000" is needed to make the change effective
```

如需进一步了解可通过此命令授予的具体权限，请参阅 Windows Azure Active Directory 权限的 GUID 表。

c. 批准权限请求。如果您的帐户没有 Azure Active Directory 租户管理员角色，请按照您的组织的准则请租户管理员批准您的权限请求。

```
$ az ad app permission grant --id <appId> \ 1
    --api 00000002-0000-0000-c000-000000000000
```

1 将 **<appId>** 替换为服务器主体的 **appId** 参数值。

3.9.3.7. 支持的 Azure 区域

安装程序会根据您的订阅动态地生成可用的 Microsoft Azure 区域列表。OpenShift Container Platform 4.6.1 中已测试并验证了以下 Azure 区域：

支持的 Azure 公共区域

- **australiacentral** (Australia Central)
- **australiaeast** (Australia East)
- **australiasoutheast** (Australia South East)
- **brazilsouth** (Brazil South)
- **canadacentral** (Canada Central)
- **canadaeast** (Canada East)
- **centralindia** (Central India)
- **centralus** (Central US)
- **eastasia** (East Asia)
- **eastus** (East US)
- **eastus2** (East US 2)
- **francecentral** (France Central)

- **germanywestcentral** (Germany West Central)
- **japaneast** (Japan East)
- **japanwest** (Japan West)
- **koreacentral** (Korea Central)
- **koreasouth** (Korea South)
- **northcentralus** (North Central US)
- **northeurope** (North Europe)
- **norwayeast** (Norway East)
- **southafricanorth** (South Africa North)
- **southcentralus** (South Central US)
- **southeastasia** (Southeast Asia)
- **southindia** (South India)
- **switzerlandnorth** (Switzerland North)
- **uaenorth** (UAE North)
- **uksouth** (UK South)
- **ukwest** (UK West)
- **westcentralus** (West Central US)
- **westeurope** (West Europe)
- **westindia** (West India)
- **westus** (West US)
- **westus2** (West US 2)

支持的 Azure 政府区域
OpenShift Container Platform 4.6 添加了对以下 Microsoft Azure Government（MAG）区域的支持：

- **usgovtexas** (US Gov Texas)
- **usgovvirginia** (US Gov Virginia)

您可以参阅 Azure 文档来了解与所有可用 MAG 区域的信息。其他 MAG 区域应该可以与 OpenShift Container Platform 一起工作，但并没有经过测试。

3.9.4. 获取安装程序

在安装 OpenShift Container Platform 之前，将安装文件下载到本地计算机上。

先决条件

- 运行 Linux 或 macOS 的计算机，本地磁盘空间为 500 MB

流程

1. 访问 OpenShift Cluster Manager 站点的 Infrastructure Provider 页面。如果您有红帽帐号，请使用自己的凭证登录。如果没有，请创建一个帐户。
2. 选择您的基础架构供应商。
3. 进入适用于您的安装类型的页面，下载您的操作系统的安装程序，并将文件放在要保存安装配置文件的目录中。。

重要

安装程序会在用来安装集群的计算机上创建若干文件。在完成集群安装后，您必须保留安装程序和安装程序所创建的文件。这两个文件都需要删除集群。

重要

删除安装程序创建的文件不会删除您的集群，即使集群在安装过程中失败也是如此。要删除集群，为特定云供应商完成 OpenShift Container Platform 卸载流程。

4. 提取安装程序。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ tar xvf openshift-install-linux.tar.gz
```

5. 从 Red Hat OpenShift Cluster Manager 下载安装 pull secret 。通过此 pull secret，您可以进行所含授权机构提供的服务的身份验证，这些服务包括为 OpenShift Container Platform 组件提供容器镜像的 Quay.io。

3.9.5. 生成 SSH 私钥并将其添加到代理中

如果要在集群上执行安装调试或灾难恢复，则必须为 **ssh-agent** 和安装程序提供 SSH 密钥。您可以使用此密钥访问公共集群中的 bootstrap 机器来排除安装问题。

注意

在生产环境中，您需要进行灾难恢复和调试。

您可以使用此密钥以 **core** 用户身份通过 SSH 连接到 master 节点。在部署集群时，此密钥会添加到 **core** 用户的 **~/.ssh/authorized_keys** 列表中。

注意

您必须使用一个本地密钥，而不要使用在特定平台上配置的密钥，如 AWS 密钥对。

流程

1. 如果还没有为计算机上免密码身份验证而配置的 SSH 密钥，请创建一个。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ ssh-keygen -t ed25519 -N '' \
  -f <path>/<file_name> 1
```

-

1 指定新 SSH 密钥的路径和文件名，如 ~/**.ssh/id_rsa**。如果您已有密钥对，请确保您的公钥位于 ~/**.ssh** 目录中。

运行此命令会在指定的位置生成不需要密码的 SSH 密钥。

注意

如果您计划在 **x86_64** 架构中安装使用 FIPS 验证的/Modules in Process 加密库的 OpenShift Container Platform 集群，不要创建使用 **ed25519** 算法的密钥。反之，创建一个使用 **rsa** 或 **ecdsa** 算法的密钥。

2. 作为后台任务启动 **ssh-agent** 进程：

```
$ eval "$(ssh-agent -s)"
```

输出示例

```
Agent pid 31874
```

注意

如果您的集群采用 FIPS 模式，则只使用 FIPS 兼容算法来生成 SSH 密钥。密钥必须是 RSA 或 ECDSA。

3. 将 SSH 私钥添加到 **ssh-agent**：

```
$ ssh-add <path>/<file_name> 1
```

输出示例

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

1 指定 SSH 私钥的路径和文件名，如 ~/**.ssh/id_rsa**

后续步骤

- 在安装 OpenShift Container Platform 时，为安装程序提供 SSH 公钥。如果在您置备的基础架构上安装集群，您必须将此密钥提供给集群的机器。

3.9.6. 创建用于 Azure 的安装文件

要使用用户置备的基础架构在 Microsoft Azure 上安装 OpenShift Container Platform，您必须生成并修改安装程序部署集群所需的文件，以便集群只创建要使用的机器。您要生成并自定义 **install-config.yaml** 文件、Kubernetes 清单和 Ignition 配置文件。您也可以选择在安装准备阶段首先设置独立的 **var** 分区。

3.9.6.1. 可选：创建独立 /var 分区

建议安装程序将 OpenShift Container Platform 的磁盘分区保留给安装程序。然而，在有些情况下您可能需要在文件系统的一部分中创建独立分区。

OpenShift Container Platform 支持添加单个分区来将存储附加到 **/var** 分区或 **/var** 的子目录。例如：

- **/var/lib/containers**：保存镜像相关的内容，随着更多镜像和容器添加到系统中，它所占用的存储会增加。
- **/var/lib/etcd**：保存您可能希望保持独立的数据，比如 etcd 存储的性能优化。
- **/var**：保存您希望独立保留的数据，用于特定目的（如审计）。

单独存储 **/var** 目录的内容可方便地根据需要对区域扩展存储，并可以在以后重新安装 OpenShift Container Platform 时保持该数据地完整。使用这个方法，您不必再次拉取所有容器，在更新系统时也无法复制大量日志文件。

因为 **/var** 在进行一个全新的 Red Hat Enterprise Linux CoreOS（RHCOS）安装前必需存在，所以这个流程会在 OpenShift Container Platform 安装过程的 **openshift-install** 准备阶段插入的机器配置来设置独立的 **/var** 分区。

重要

如果按照以下步骤在此流程中创建独立 **/var** 分区，则不需要再次创建 Kubernetes 清单和 Ignition 配置文件，如本节所述。

流程

1. 创建存放 OpenShift Container Platform 安装文件的目录：

```
$ mkdir $HOME/clusterconfig
```

2. 运行 **openshift-install** 在 **manifest** 和 **openshift** 子目录中创建一组文件。在出现提示时回答系统问题：

```
$ openshift-install create manifests --dir $HOME/clusterconfig
```

输出示例

```
? SSH Public Key ...
INFO Credentials loaded from the "myprofile" profile in file "/home/myuser/.aws/credentials"
INFO Consuming Install Config from target directory
INFO Manifests created in: $HOME/clusterconfig/manifests and
$HOME/clusterconfig/openshift
```

3. 可选：确认安装程序在 **clusterconfig/openshift** 目录中创建了清单：

```
$ ls $HOME/clusterconfig/openshift/
```

输出示例

```
99_kubeadmin-password-secret.yaml
99_openshift-cluster-api_master-machines-0.yaml
99_openshift-cluster-api_master-machines-1.yaml
99_openshift-cluster-api_master-machines-2.yaml
...
```

4. 创建 **MachineConfig** 对象并将其添加到 **openshift** 目录中的一个文件中。例如，把文件命名为 **98-var-partition.yaml**，将磁盘设备名称改为 **worker** 系统中存储设备的名称，并根据情况设置存储大小。这个示例将 **/var** 目录放在一个单独的分区中：

```
apiVersion: machineconfiguration.openshift.io/v1
kind: MachineConfig
metadata:
  labels:
    machineconfiguration.openshift.io/role: worker
  name: 98-var-partition
spec:
  config:
    ignition:
      version: 3.1.0
    storage:
      disks:
      - device: /dev/<device_name> 1
        partitions:
        - label: var
          startMiB: <partition_start_offset> 2
          sizeMiB: <partition_size> 3
      filesystems:
        - device: /dev/disk/by-partlabel/var
          path: /var
          format: xfs
    systemd:
      units:
        - name: var.mount 4
          enabled: true
          contents: |
            [Unit]
            Before=local-fs.target
            [Mount]
            What=/dev/disk/by-partlabel/var
            Where=/var
            Options=defaults,prjquota 5
            [Install]
            WantedBy=local-fs.target
```

1. 要分区的磁盘的存储设备名称。
2. 当在引导磁盘中添加数据分区时，推荐最少使用 25000 MiB（Mebibytes）。root 文件系统会自动重新定义大小使其占据所有可用空间（最多到指定的偏移值）。如果没有指定值，或者指定的值小于推荐的最小值，则生成的 root 文件系统会太小，而在以后进行的 RHCOS 重新安装可能会覆盖数据分区的开始部分。
3. 数据分区的大小（以兆字节为单位）。
4. 挂载单元的名称必须与 **Where=** 指令中指定的目录匹配。例如，对于挂载在 **/var/lib/containers** 上的文件系统，该单元必须命名为 **var-lib-containers.mount**。
5. 对于用于容器存储的文件系统，必须启用 **prjquota** 挂载选项。

注意

在创建独立 **/var** 分区时，如果不同的实例类型没有相同的设备名称，则无法将不同的实例类型用于 worker 节点。

5. 再次运行 **openshift-install**，从 **manifest** 和 **openshift** 子目录中的一组文件创建 Ignition 配置：

```
$ openshift-install create ignition-configs --dir $HOME/clusterconfig
$ ls $HOME/clusterconfig/
auth  bootstrap.ign  master.ign  metadata.json  worker.ign
```

现在，您可以使用 Ignition 配置文件作为安装程序的输入来安装 Red Hat Enterprise Linux CoreOS（RHCOS）系统。

3.9.6.2. 创建安装配置文件

您可以自定义在 Microsoft Azure 上安装的 OpenShift Container Platform 集群。

先决条件

- 获取 OpenShift Container Platform 安装程序以及集群的 pull secret。

流程

1. 创建 **install-config.yaml** 文件。
   a. 更改到包含安装程序的目录，再运行以下命令：

```
$ ./openshift-install create install-config --dir <installation_directory> 1
```

1 对于 **<installation_directory>**，请指定用于保存安装程序所创建的文件的目录名称。

重要

指定一个空目录。一些安装信息，如 bootstrap X.509 证书，有较短的过期间隔，因此不要重复使用安装目录。如果要重复使用另一个集群安装中的个别文件，可以将其复制到您的目录中。但是，一些安装数据的文件名可能会在发行版本之间有所改变。从 OpenShift Container Platform 老版本中复制安装文件时要格外小心。

   b. 在提示符处，提供您的云的配置详情：
      i. 可选：选择用来访问集群机器的 SSH 密钥。

注意

对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 **ssh-agent** 进程使用的 SSH 密钥。

      ii. 选择 **azure** 作为目标平台。

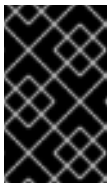
iii. 如果计算机上没有 Microsoft Azure 配置集，请为您的订阅和服务主体指定以下 Azure 参数值：

- **azure subscription id**：要用于集群的订阅 ID。指定帐户输出中的 **id** 值。
- **azure tenant id**：租户 ID。指定帐户输出中的 **tenantId** 值。
- **azure service principal client id**：服务主体的 **appId** 参数值。
- **azure service principal client secret**：服务主体的 **password** 参数值。

iv. 选择要在其中部署集群的区域。

v. 选择集群要部署到的基域。基域与您为集群创建的 Azure DNS 区对应。

vi. 为集群输入一个描述性名称。



重要

所有通过公共端点提供的 Azure 资源均存在资源名称的限制，您无法创建使用某些名称的资源。如需 Azure 限制词语列表，请参阅 Azure 文档中的解决保留资源名称错误。

vii. 粘贴 Red Hat OpenShift Cluster Manager 中的 pull secret 。

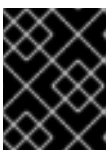
c. 可选：如果您不希望集群置备计算机器，请通过编辑 **install-config.yaml** 文件将 **compute** 池的 **replicas** 设置为 **0** 来清空计算池。

```
compute:
- hyperthreading: Enabled
  name: worker
  platform: {}
  replicas: 0 1
```

1 设置为 **0**。

2. 修改 **install-config.yaml** 文件。您可以在**安装配置参数**部分中找到有关可用参数的更多信息。

3. 备份 **install-config.yaml** 文件，以便用于安装多个集群。



重要

install-config.yaml 文件会在安装过程中消耗掉。如果要重复使用此文件，必须现在备份。

3.9.6.3. 在安装过程中配置集群范围代理

生产环境可能会拒绝直接访问互联网，而是提供 HTTP 或 HTTPS 代理。您可以通过在 **install-config.yaml** 文件中配置代理设置，将新的 OpenShift Container Platform 集群配置为使用代理。

先决条件

- 您有一个现有的 **install-config.yaml** 文件。

- 您检查了集群需要访问的站点，并决定是否需要绕过代理。默认情况下代理所有集群出口流量，包括对托管云供应商 API 的调用。您需要将站点添加到 **Proxy** 对象的 **spec.noProxy** 字段来绕过代理。

注意

Proxy 对象 **status.noProxy** 字段使用安装配置中的 **networking.machineNetwork[].cidr**、**networking.clusterNetwork[].cidr** 和 **networking.serviceNetwork[]** 字段的值填充。

对于在 Amazon Web Services(AWS)、Google Cloud Platform(GCP)、Microsoft Azure 和 Red Hat OpenStack Platform(RHOSP)上安装，**Proxy** 对象 **status.noProxy** 字段也会使用实例元数据端点填充(**169.254.169.254**)。

流程

1. 编辑 **install-config.yaml** 文件并添加代理设置。例如：

```
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: example.com 3
additionalTrustBundle: | 4
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
...
```

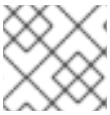
1. 用于创建集群外 HTTP 连接的代理 URL。URL 必须是 **http**。
2. 用于创建集群外 HTTPS 连接的代理 URL。
3. 要排除在代理中的目标域名、IP 地址或其他网络 CIDR 的逗号分隔列表。在域前面加 **.** 来仅匹配子域。例如：**.y.com** 匹配 **x.y.com**，但不匹配 **y.com**。使用 ***** 绕过所有目的地的代理。
4. 如果提供，安装程序会在 **openshift-config 命名空间中生成名为 user-ca- bundle** 的配置映射来保存额外的 CA 证书。如果您提供 **additionalTrustBundle** 和至少一个代理设置，则 **Proxy** 对象会被配置为引用 **trustedCA** 字段中的 **user-ca-bundle** 配置映射。然后，Cluster Network Operator 会创建一个 **trusted-ca-bundle** 配置映射，该配置映射将为 **trustedCA** 参数指定的内容与 RHCOS 信任捆绑包合并。**additionalTrustBundle** 字段是必需的，除非代理的身份证书由来自 RHCOS 信任捆绑包的颁发机构签名。

注意

安装程序不支持代理的 **readinessEndpoints** 字段。

2. 保存该文件，并在安装 OpenShift Container Platform 时引用。

安装程序会创建一个名为 **cluster** 的集群范围代理，该代理使用提供的 **install-config.yaml** 文件中的代理设置。如果没有提供代理设置，仍然会创建一个 **cluster Proxy** 对象，但它会有一个空 **spec**。



注意

只支持名为 **cluster** 的 **Proxy** 对象，且无法创建额外的代理。

3.9.6.4. 为 ARM 模板导出常用变量

您必须导出与提供的 Azure Resource Manager（ARM）模板搭配使用的一组常用变量，它们有助于在 Microsoft Azure 上完成用户提供基础架构安装。

注意

特定的 ARM 模板可能还需要其他导出变量，这些变量在相关的程序中详细介绍。

先决条件

- 获取 OpenShift Container Platform 安装程序以及集群的 pull secret。

流程

1. 导出 **install-config.yaml** 中由提供的 ARM 模板使用的通用变量：

```
$ export CLUSTER_NAME=<cluster_name> (1)
$ export AZURE_REGION=<azure_region> (2)
$ export SSH_KEY=<ssh_key> (3)
$ export BASE_DOMAIN=<base_domain> (4)
$ export BASE_DOMAIN_RESOURCE_GROUP=<base_domain_resource_group> (5)
```

(1) **install-config.yaml** 文件中的**.metadata.name** 属性的值。

(2) 集群要部署到的区域，如 **centralus** 。这是来自 **install-config.yaml** 文件中的 **.platform.azure.region** 属性的值。

(3) 作为字符串的 SSH RSA 公钥文件。您必须使用引号包括 SSH 密钥，因为它包含空格。这是 **install-config.yaml 文件中的 .sshKey** 属性的值。

(4) 集群要部署到的基域。基域与您为集群创建的公共 DNS 区对应。这是 **install-config.yaml 文件中的 .baseDomain** 属性的值。

(5) 公共 DNS 区所在的资源组。这是 **install-config.yaml 文件中的 .platform.azure.baseDomainResourceGroupName** 属性的值。

例如：

```
$ export CLUSTER_NAME=test-cluster
$ export AZURE_REGION=centralus
$ export SSH_KEY="ssh-rsa xxx/xxx/xxx= user@email.com"
$ export BASE_DOMAIN=example.com
$ export BASE_DOMAIN_RESOURCE_GROUP=ocp-cluster
```

2. 导出 kubeadmin 凭证：

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig (1)
```

1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

3.9.6.5. 创建 Kubernetes 清单和 Ignition 配置文件

由于您必须修改一些集群定义文件并要手动启动集群机器，因此您必须生成 Kubernetes 清单和 Ignition 配置文件，集群需要这两项来创建其机器。

安装配置文件转换为 Kubernetes 清单。清单嵌套到 Ignition 配置文件中，稍后用于创建集群。

重要

- 安装程序生成的 Ignition 配置文件包含在 24 小时后过期的证书，然后在过期时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外情况是，您需要手动批准待处理的 **node-bootstrapper** 证书签名请求（CSR）来恢复 kubelet 证书。如需更多信息，请参阅*从过期的 control plane 证书中恢复*的文档。
- 建议您在生成 12 小时后使用 Ignition 配置文件，因为集群安装后 24 小时证书从 16 小时轮转至 22 小时。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中运行证书更新时避免安装失败。

先决条件

- 已获得 OpenShift Container Platform 安装程序。
- 已创建 **install-config.yaml** 安装配置文件。

流程

1. 切换到包含安装程序的目录，并为集群生成 Kubernetes 清单：

```
$ ./openshift-install create manifests --dir <installation_directory> 1
```

1 对于 **<installation_directory>**，请指定含有您创建的 **install-config.yaml** 文件的安装目录。

2. 删除定义 control plane 机器的 Kubernetes 清单文件：

```
$ rm -f <installation_directory>/openshift/99_openshift-cluster-api_master-machines-*.yaml
```

通过删除这些文件，您可以防止集群自动生成 control plane 机器。

3. 删除定义 worker 机器的 Kubernetes 清单文件：

```
$ rm -f <installation_directory>/openshift/99_openshift-cluster-api_worker-machineset-*.yaml
```

由于您要自行创建并管理 worker 机器，因此不需要初始化这些机器。

4. 检查 **<installation_directory>/manifests/cluster-scheduler-02-config.yml** Kubernetes 清单文件中的 **mastersSchedulable** 参数是否已设置为 **false**。此设置可防止在 control plane 机器上调度 pod：

   a. 打开 **<installation_directory>/manifests/cluster-scheduler-02-config.yml** 文件。

b. 找到 **mastersSchedulable** 参数并确保它被设置为 **false**。

c. 保存并退出文件。

5. 可选：如果您不希望 Ingress Operator 代表您创建 DNS 记录，请删除 **<installation_directory>/manifests/cluster-dns-02-config.yml** DNS 配置文件中的 **privateZone** 和 **publicZone** 部分：

```
apiVersion: config.openshift.io/v1
kind: DNS
metadata:
  creationTimestamp: null
  name: cluster
spec:
  baseDomain: example.openshift.com
  privateZone: 1
    id: mycluster-100419-private-zone
  publicZone: 2
    id: example.openshift.com
status: {}
```

1 2 完全删除此部分。

如果您这样做，后续步骤中必须手动添加入口 DNS 记录。

6. 在用户置备的基础架构上配置 Azure 时，您必须导出清单文件中定义的一些常见变量，以备稍后在 Azure Resource Manager（ARM）模板中使用：

a. 使用以下命令导出基础架构 ID:

```
$ export INFRA_ID=<infra_id> 1
```

1 OpenShift Container Platform 集群被分配了一个标识符（**INFRA_ID**），其格式为 **<cluster_name>-<random_string>**。这将作为使用提供的 ARM 模板创建的大部分资源的基本名称。这是 **manifests/cluster-infrastructure-02-config.yml 文件中的 .status.infrastructureName** 属性的值。

b. 使用以下命令导出资源组：

```
$ export RESOURCE_GROUP=<resource_group> 1
```

1 此 Azure 部署中创建的所有资源都作为资源组的一部分。资源组名称还基于 **INFRA_ID**，格式为 **<cluster_name>-<random_string>-rg**。这是 **manifests/cluster-infrastructure-02-config.yml 文件中的 .status.platformStatus.azure.resourceGroupName** 属性的值。

7. 要创建 Ignition 配置文件，从包含安装程序的目录运行以下命令：

```
$ ./openshift-install create ignition-configs --dir <installation_directory> 1
```

1 对于 **<installation_directory>**，请指定相同的安装目录。

该目录中将生成以下文件：

```
.
├── auth
│   ├── kubeadmin-password
│   └── kubeconfig
├── bootstrap.ign
├── master.ign
├── metadata.json
└── worker.ign
```

3.9.7. 创建 Azure 资源组和身份

您必须创建一个 Microsoft Azure 资源组以及该资源组的身份。它们都用于在 Azure 上安装 OpenShift Container Platform 集群。

先决条件

- 配置 Azure 帐户。
- 为集群生成 Ignition 配置文件。

流程

1. 在受支持的 Azure 区域中创建资源组：

```
$ az group create --name ${RESOURCE_GROUP} --location ${AZURE_REGION}
```

2. 为资源组创建 Azure 身份：

```
$ az identity create -g ${RESOURCE_GROUP} -n ${INFRA_ID}-identity
```

这用于授予集群中 Operator 所需的访问权限。例如，这允许 Ingress Operator 创建公共 IP 及其负载均衡器。您必须将 Azure 身份分配给角色。

3. 将 Contributor 角色授予 Azure 身份：

   a. 导出 Azure 角色分配所需的以下变量：

```
$ export PRINCIPAL_ID=`az identity show -g ${RESOURCE_GROUP} -n ${INFRA_ID}-identity --query principalId --out tsv`
```

```
$ export RESOURCE_GROUP_ID=`az group show -g ${RESOURCE_GROUP} --query id --out tsv`
```

   b. 将 Contributor 角色分配给身份：

```
$ az role assignment create --assignee "${PRINCIPAL_ID}" --role 'Contributor' --scope "${RESOURCE_GROUP_ID}"
```

3.9.8. 上传 RHCOS 集群镜像和 bootstrap Ignition 配置文件

Azure 客户端不支持基于本地现有文件进行的部署，因此您必须复制 RHCOS 虚拟硬盘（VHD）集群镜像，并将 bootstrap Ignition 配置文件存储在存储容器中，以便在部署过程中访问这些文件。

先决条件

- 配置 Azure 帐户。
- 为集群生成 Ignition 配置文件。

流程

1. 创建 Azure 存储帐户以存储 VHD 集群镜像：

```
$ az storage account create -g ${RESOURCE_GROUP} --location ${AZURE_REGION} --
name ${CLUSTER_NAME}sa --kind Storage --sku Standard_LRS
```

警告

Azure 存储帐户名称的长度必须在 3 到 24 个字符之，且只使用数字和小写字母。如果您的 **CLUSTER_NAME** 变量没有遵循这些限制，您必须手动定义 Azure 存储帐户名称。如需有关 Azure 存储帐户名称限制的更多信息，请参阅 Azure 文档中的解决存储帐户名称的错误。

2. 将存储帐户密钥导出为环境变量：

```
$ export ACCOUNT_KEY=`az storage account keys list -g ${RESOURCE_GROUP} --
account-name ${CLUSTER_NAME}sa --query "[0].value" -o tsv`
```

3. 选择 RHCOS 版本以使用并导出 VHD 的 URL 到环境变量：

```
$ export VHD_URL=`curl -s https://raw.githubusercontent.com/openshift/installer/release-
4.6/data/data/rhcos.json | jq -r .azure.url`
```

重要

RHCOS 镜像可能不会随着 OpenShift Container Platform 的每一发行版本都有改变。您必须指定一个最高版本的镜像，其版本号应小于或等于您安装的 OpenShift Container Platform 版本。如果可用，请使用与 OpenShift Container Platform 版本匹配的镜像版本。

4. 将所选 VHD 复制到 blob:

```
$ az storage container create --name vhd --account-name ${CLUSTER_NAME}sa --account-
key ${ACCOUNT_KEY}
```

```
$ az storage blob copy start --account-name ${CLUSTER_NAME}sa --account-key
${ACCOUNT_KEY} --destination-blob "rhcos.vhd" --destination-container vhd --source-uri
"${VHD_URL}"
```

–

要跟踪 VHD 复制任务的进程，请运行这个脚本：

```
status="unknown"
while [ "$status" != "success" ]
do
  status=`az storage blob show --container-name vhd --name "rhcos.vhd" --account-name
${CLUSTER_NAME}sa --account-key ${ACCOUNT_KEY} -o tsv --query
properties.copy.status`
  echo $status
done
```

5. 创建 blob 存储容器并上传生成的 **bootstrap.ign** 文件：

```
$ az storage container create --name files --account-name ${CLUSTER_NAME}sa --
account-key ${ACCOUNT_KEY} --public-access blob
```

```
$ az storage blob upload --account-name ${CLUSTER_NAME}sa --account-key
${ACCOUNT_KEY} -c "files" -f "<installation_directory>/bootstrap.ign" -n "bootstrap.ign"
```

3.9.9. 创建 DNS 区示例

使用用户置备的基础架构的集群需要 DNS 记录。您应该选择适合您的场景的 DNS 策略。

在本例中，使用了 Azure 的 DNS 解决方案，因此您将为外部（内部网络）可见性创建一个新的公共 DNS 区域，并为内部集群解析创建一个私有 DNS 区域。

注意

公共 DNS 区域不需要与集群部署位于同一个资源组中，且可能已在您的机构中为所需基域存在。如果情况如此，您可以跳过创建公共 DNS 区这一步 ; 请确定您之前生成的安装配置反映了这种情况。

先决条件

- 配置 Azure 帐户。
- 为集群生成 Ignition 配置文件。

流程

1. 在 **BASE_DOMAIN_RESOURCE_GROUP** 环境变量中导出的资源组中创建新的公共 DNS 区域：

```
$ az network dns zone create -g ${BASE_DOMAIN_RESOURCE_GROUP} -n
${CLUSTER_NAME}.${BASE_DOMAIN}
```

如果您使用的是公共 DNS 区域，可以跳过这一步。

2. 在与这个部署的其余部分相同的资源组中创建私有 DNS 区域：

```
$ az network private-dns zone create -g ${RESOURCE_GROUP} -n
${CLUSTER_NAME}.${BASE_DOMAIN}
```

如需了解更多信息，请参阅在 Azure 中配置公共 DNS 的信息。

3.9.10. 在 Azure 中创建 VNet

您必须在 Microsoft Azure 中创建虚拟网络（VNet），供您的 OpenShift Container Platform 集群使用。您可以对 VNet 进行定制来满足您的要求。创建 VNet 的一种方法是修改提供的 Azure Resource Manager（ARM）模板。

注意

如果不使用提供的 ARM 模板来创建 Azure 基础架构，您必须检查提供的信息并手动创建基础架构。如果集群没有正确初始化，您可能需要联系红帽支持并提供您的安装日志。

先决条件

- 配置 Azure 帐户。
- 为集群生成 Ignition 配置文件。

流程

1. 复制 **VNet 的 ARM 模板** 一节中的模板，并将它以 **01_vnet.json** 保存到集群的安装目录中。此模板描述了集群所需的 VNet。
2. 使用 **az** CLI 创建部署：

```
$ az deployment group create -g ${RESOURCE_GROUP} \
  --template-file "<installation_directory>/01_vnet.json" \
  --parameters baseName="${INFRA_ID}" 1
```

1 资源名称使用的基本名称；这通常是集群的基础架构 ID。

3. 将 VNet 模板链接到私有 DNS 区域：

```
$ az network private-dns link vnet create -g ${RESOURCE_GROUP} -z
${CLUSTER_NAME}.${BASE_DOMAIN} -n ${INFRA_ID}-network-link -v "${INFRA_ID}-vnet"
-e false
```

3.9.10.1. VNet 的 ARM 模板

您可以使用以下 Azure Resource Manager（ARM）模板来部署 OpenShift Container Platform 集群所需的 VPC：

例 3.1. 01_vnet.json ARM 模板

```
{
  "$schema" : "https://schema.management.azure.com/schemas/2015-01-
01/deploymentTemplate.json#",
  "contentVersion" : "1.0.0.0",
  "parameters" : {
    "baseName" : {
      "type" : "string",
      "minLength" : 1,
```

```
    "metadata" : {
      "description" : "Base name to be used in resource names (usually the cluster's Infra ID)"
    }
   }
  },
  "variables" : {
   "location" : "[resourceGroup().location]",
   "virtualNetworkName" : "[concat(parameters('baseName'), '-vnet')]",
   "addressPrefix" : "10.0.0.0/16",
   "masterSubnetName" : "[concat(parameters('baseName'), '-master-subnet')]",
   "masterSubnetPrefix" : "10.0.0.0/24",
   "nodeSubnetName" : "[concat(parameters('baseName'), '-worker-subnet')]",
   "nodeSubnetPrefix" : "10.0.1.0/24",
   "clusterNsgName" : "[concat(parameters('baseName'), '-nsg')]"
  },
  "resources" : [
   {
    "apiVersion" : "2018-12-01",
    "type" : "Microsoft.Network/virtualNetworks",
    "name" : "[variables('virtualNetworkName')]",
    "location" : "[variables('location')]",
    "dependsOn" : [
     "[concat('Microsoft.Network/networkSecurityGroups/', variables('clusterNsgName'))]"
    ],
    "properties" : {
     "addressSpace" : {
      "addressPrefixes" : [
       "[variables('addressPrefix')]"
      ]
     },
     "subnets" : [
      {
       "name" : "[variables('masterSubnetName')]",
       "properties" : {
        "addressPrefix" : "[variables('masterSubnetPrefix')]",
        "serviceEndpoints": [],
        "networkSecurityGroup" : {
         "id" : "[resourceId('Microsoft.Network/networkSecurityGroups',
variables('clusterNsgName'))]"
        }
       }
      },
      {
       "name" : "[variables('nodeSubnetName')]",
       "properties" : {
        "addressPrefix" : "[variables('nodeSubnetPrefix')]",
        "serviceEndpoints": [],
        "networkSecurityGroup" : {
         "id" : "[resourceId('Microsoft.Network/networkSecurityGroups',
variables('clusterNsgName'))]"
        }
       }
      }
     ]
    }
   },
```

```
    {
      "type" : "Microsoft.Network/networkSecurityGroups",
      "name" : "[variables('clusterNsgName')]",
      "apiVersion" : "2018-10-01",
      "location" : "[variables('location')]",
      "properties" : {
        "securityRules" : [
          {
            "name" : "apiserver_in",
            "properties" : {
              "protocol" : "Tcp",
              "sourcePortRange" : "*",
              "destinationPortRange" : "6443",
              "sourceAddressPrefix" : "*",
              "destinationAddressPrefix" : "*",
              "access" : "Allow",
              "priority" : 101,
              "direction" : "Inbound"
            }
          }
        ]
      }
    }
  ]
}
```

3.9.11. 为 Azure 基础架构创建 RHCOS 集群镜像

您必须对 OpenShift Container Platform 节点的 Microsoft Azure 使用有效的 Red Hat Enterprise Linux CoreOS（RHCOS）镜像。

先决条件

- 配置 Azure 帐户。
- 为集群生成 Ignition 配置文件。
- 将 RHCOS 虚拟硬盘（VHD）集群镜像存储在 Azure 存储容器中。
- 在 Azure 存储容器中存储 bootstrap Ignition 配置文件。

流程

1. 复制**镜像存储的 ARM 模板** 部分中的模板，并将它以 **02_storage.json** 保存到集群的安装目录中。此模板描述了集群所需的镜像存储。
2. 以一个变量的形式将 RHCOS VHD blob URL 导出：

```
$ export VHD_BLOB_URL=`az storage blob url --account-name ${CLUSTER_NAME}sa --
account-key ${ACCOUNT_KEY} -c vhd -n "rhcos.vhd" -o tsv`
```

3. 部署集群镜像

```
$ az deployment group create -g ${RESOURCE_GROUP} \
  --template-file "<installation_directory>/02_storage.json" \
  --parameters vhdBlobURL="${VHD_BLOB_URL}" \ 1
  --parameters baseName="${INFRA_ID}" 2
```

1 用于创建 master 和 worker 机器的 RHCOS VHD 的 blob URL。

2 资源名称使用的基本名称 ; 这通常是集群的基础架构 ID。

3.9.11.1. 镜像存储的 ARM 模板

您可以使用以下 Azure Resource Manager（ARM）模板来部署 OpenShift Container Platform 集群所需的存储的 Red Hat Enterprise Linux CoreOS（RHCOS）镜像：

例 3.2. 02_storage.json ARM 模板

```
{
  "$schema" : "https://schema.management.azure.com/schemas/2015-01-01/deploymentTemplate.json#",
  "contentVersion" : "1.0.0.0",
  "parameters" : {
    "baseName" : {
      "type" : "string",
      "minLength" : 1,
      "metadata" : {
        "description" : "Base name to be used in resource names (usually the cluster's Infra ID)"
      }
    },
    "vhdBlobURL" : {
      "type" : "string",
      "metadata" : {
        "description" : "URL pointing to the blob where the VHD to be used to create master and worker machines is located"
      }
    }
  },
  "variables" : {
    "location" : "[resourceGroup().location]",
    "imageName" : "[concat(parameters('baseName'), '-image')]"
  },
  "resources" : [
    {
      "apiVersion" : "2018-06-01",
      "type": "Microsoft.Compute/images",
      "name": "[variables('imageName')]",
      "location" : "[variables('location')]",
      "properties": {
        "storageProfile": {
          "osDisk": {
            "osType": "Linux",
            "osState": "Generalized",
            "blobUri": "[parameters('vhdBlobURL')]",
            "storageAccountType": "Standard_LRS"
          }
```

```
        }
      }
    }
  ]
}
```

3.9.12. 用户置备的基础架构对网络的要求

所有 Red Hat Enterprise Linux CoreOS（RHCOS）机器在启动过程中需要 **initramfs** 中的网络从机器配置服务器获取 Ignition 配置。

您必须配置机器间的网络连接，以便集群组件进行通信。每台机器都必须能够解析集群中所有其他机器的主机名。

表 3.29. 所有机器到所有机器

协议	端口	描述
ICMP	N/A	网络可访问性测试
TCP	**1936**	指标
	9000-9999	主机级别的服务，包括端口 **9100-9101** 上的节点导出器和端口 **9099** 上的 Cluster Version Operator。
	10250-10259	Kubernetes 保留的默认端口
	10256	openshift-sdn
UDP	**4789**	VXLAN 和 Geneve
	6081	VXLAN 和 Geneve
	9000-9999	主机级别的服务，包括端口 **9100-9101** 上的节点导出器。
TCP/UDP	**30000-32767**	Kubernetes 节点端口

表 3.30. 要通过控制平面的所有机器

协议	端口	描述
TCP	**6443**	Kubernetes API

表 3.31. control plane 机器到 control plane 机器

协议	端口	描述
TCP	**2379-2380**	etcd 服务器和对等端口

网络拓扑要求

您为集群置备的基础架构必须满足下列网络拓扑要求。

重要

OpenShift Container Platform 要求所有节点都能访问互联网，以便为平台容器提取镜像并向红帽提供遥测数据。

负载均衡器

在安装 OpenShift Container Platform 前，您必须置备两个满足以下要求的负载均衡器：

1. **API 负载均衡器**：提供一个通用端点，供用户（包括人和机器）与平台交互和配置。配置以下条件：
   - 只适用于第 4 层负载均衡。这可被称为 Raw TCP、SSL Passthrough 或者 SSL 桥接模式。如果使用 SSL Bridge 模式，必须为 API 路由启用 Server Name Indication（SNI）。
   - 无状态负载平衡算法。这些选项根据负载均衡器的实现而有所不同。

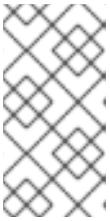
重要

不要为 API 负载均衡器配置会话持久性。

在负载均衡器的前端和后台配置以下端口：

表 3.32. API 负载均衡器

端口	后端机器（池成员）	内部	外部	描述
6443	Bootstrap 和 control plane.bootstrap 机器初始化集群 control plane 后，您要从负载均衡器中删除 bootstrap 机器。您必须为 API 服务器健康检查探测配置 **/readyz** 端点。	X	X	Kubernetes API 服务器
22623	Bootstrap 和 control plane.bootstrap 机器初始化集群 control plane 后，您要从负载均衡器中删除 bootstrap 机器。	X		机器配置服务器



注意

负载均衡器必须配置为，从 API 服务器关闭 **/readyz** 端点到从池中删除 API 服务器实例时最多需要 30 秒。在 **/readyz** 返回错误或处于健康状态后的时间范围内，端点必须被删除或添加。每 5 秒或 10 秒探测一次，有两个成功请求处于健康状态，三个成为不健康的请求经过测试。

2. **应用程序入口负载均衡器**:提供来自集群外部的应用程序流量流量的 Ingress 点。配置以下条件：

   - 只适用于第 4 层负载均衡。这可被称为 Raw TCP、SSL Passthrough 或者 SSL 桥接模式。如果使用 SSL Bridge 模式，您必须为 Ingress 路由启用Server Name Indication（SNI）。
   - 建议根据可用选项以及平台上托管的应用程序类型，使用基于连接的或者基于会话的持久性。

   在负载均衡器的前端和后台配置以下端口：

表 3.33. 应用程序入口负载均衡器

端口	后端机器（池成员）	内部	外部	描述
443	默认运行入口路由器 Pod、计算或 worker 的机器。	X	X	HTTPS 流量
80	默认运行入口路由器 Pod、计算或 worker 的机器。	X	X	HTTP 流量

提示

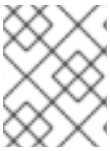
如果负载均衡器可以看到客户端的真实 IP 地址，启用基于 IP 的会话持久性可提高使用端到端 TLS 加密的应用程序的性能。

注意

OpenShift Container Platform 集群需要正确配置入口路由器。control plane 初始化后，您必须配置入口路由器。

3.9.13. 在 Azure 中创建网络和负载均衡组件

您必须在 Microsoft Azure 中配置网络和负载均衡，供您的 OpenShift Container Platform 集群使用。创建这些组件的一种方法是修改提供的 Azure Resource Manager（ARM）模板。



注意

如果不使用提供的 ARM 模板来创建 Azure 基础架构，您必须检查提供的信息并手动创建基础架构。如果集群没有正确初始化，您可能需要联系红帽支持并提供您的安装日志。

先决条件

- 配置 Azure 帐户。
- 为集群生成 Ignition 配置文件。
- 在 Azure 中创建和配置 VNet 及相关子网。

流程

1. 复制**网络和负载均衡器的 ARM 模板**一节中的模板，并将以 **03_infra.json** 保存到集群的安装目录中。此模板描述了集群所需的网络和负载均衡对象。

2. 使用 **az** CLI 创建部署：

```
$ az deployment group create -g ${RESOURCE_GROUP} \
  --template-file "<installation_directory>/03_infra.json" \
  --parameters privateDNSZoneName="${CLUSTER_NAME}.${BASE_DOMAIN}" \ 1
  --parameters baseName="${INFRA_ID}" 2
```

1. 私有 DNS 区的名称。
2. 资源名称使用的基本名称；这通常是集群的基础架构 ID。

3. 在公共区为 API 公共负载均衡器创建一个 **api** DNS 记录。**${BASE_DOMAIN_RESOURCE_GROUP}** 变量必须指向存在公共 DNS 区的资源组。

   a. 导出以下变量：

```
$ export PUBLIC_IP=`az network public-ip list -g ${RESOURCE_GROUP} --query "[?
name=='${INFRA_ID}-master-pip'] | [0].ipAddress" -o tsv`
```

   b. 在一个新的公共区中创建 DNS 记录：

```
$ az network dns record-set a add-record -g ${BASE_DOMAIN_RESOURCE_GROUP} -
z ${CLUSTER_NAME}.${BASE_DOMAIN} -n api -a ${PUBLIC_IP} --ttl 60
```

   c. 如果您要将集群添加到现有的公共区，您可以在其中创建 DNS 记录：

```
$ az network dns record-set a add-record -g ${BASE_DOMAIN_RESOURCE_GROUP} -
z ${BASE_DOMAIN} -n api.${CLUSTER_NAME} -a ${PUBLIC_IP} --ttl 60
```

3.9.13.1. 网络和负载均衡器的 ARM 模板

您可以使用以下 Azure Resource Manager（ARM）模板来部署 OpenShift Container Platform 集群所需的网络对象和负载均衡器：

例 3.3. 03_infra.json ARM 模板

```
{
  "$schema" : "https://schema.management.azure.com/schemas/2015-01-
01/deploymentTemplate.json#",
  "contentVersion" : "1.0.0.0",
  "parameters" : {
    "baseName" : {
      "type" : "string",
      "minLength" : 1,
      "metadata" : {
        "description" : "Base name to be used in resource names (usually the cluster's Infra ID)"
      }
    },
    "privateDNSZoneName" : {
      "type" : "string",
      "metadata" : {
        "description" : "Name of the private DNS zone"
      }
```

```
    }
  },
  "variables" : {
    "location" : "[resourceGroup().location]",
    "virtualNetworkName" : "[concat(parameters('baseName'), '-vnet')]",
    "virtualNetworkID" : "[resourceId('Microsoft.Network/virtualNetworks',
variables('virtualNetworkName'))]",
    "masterSubnetName" : "[concat(parameters('baseName'), '-master-subnet')]",
    "masterSubnetRef" : "[concat(variables('virtualNetworkID'), '/subnets/',
variables('masterSubnetName'))]",
    "masterPublicIpAddressName" : "[concat(parameters('baseName'), '-master-pip')]",
    "masterPublicIpAddressID" : "[resourceId('Microsoft.Network/publicIPAddresses',
variables('masterPublicIpAddressName'))]",
    "masterLoadBalancerName" : "[concat(parameters('baseName'), '-public-lb')]",
    "masterLoadBalancerID" : "[resourceId('Microsoft.Network/loadBalancers',
variables('masterLoadBalancerName'))]",
    "internalLoadBalancerName" : "[concat(parameters('baseName'), '-internal-lb')]",
    "internalLoadBalancerID" : "[resourceId('Microsoft.Network/loadBalancers',
variables('internalLoadBalancerName'))]",
    "skuName": "Standard"
  },
  "resources" : [
    {
      "apiVersion" : "2018-12-01",
      "type" : "Microsoft.Network/publicIPAddresses",
      "name" : "[variables('masterPublicIpAddressName')]",
      "location" : "[variables('location')]",
      "sku": {
        "name": "[variables('skuName')]"
      },
      "properties" : {
        "publicIPAllocationMethod" : "Static",
        "dnsSettings" : {
          "domainNameLabel" : "[variables('masterPublicIpAddressName')]"
        }
      }
    },
    {
      "apiVersion" : "2018-12-01",
      "type" : "Microsoft.Network/loadBalancers",
      "name" : "[variables('masterLoadBalancerName')]",
      "location" : "[variables('location')]",
      "sku": {
        "name": "[variables('skuName')]"
      },
      "dependsOn" : [
        "[concat('Microsoft.Network/publicIPAddresses/', variables('masterPublicIpAddressName'))]"
      ],
      "properties" : {
        "frontendIPConfigurations" : [
          {
            "name" : "public-lb-ip",
            "properties" : {
              "publicIPAddress" : {
                "id" : "[variables('masterPublicIpAddressID')]"
              }
```

```
          }
        }
      ],
      "backendAddressPools" : [
        {
          "name" : "public-lb-backend"
        }
      ],
      "loadBalancingRules" : [
        {
          "name" : "api-internal",
          "properties" : {
            "frontendIPConfiguration" : {
              "id" :"[concat(variables('masterLoadBalancerID'), '/frontendIPConfigurations/public-lb-
ip')]"
            },
            "backendAddressPool" : {
              "id" : "[concat(variables('masterLoadBalancerID'), '/backendAddressPools/public-lb-
backend')]"
            },
            "protocol" : "Tcp",
            "loadDistribution" : "Default",
            "idleTimeoutInMinutes" : 30,
            "frontendPort" : 6443,
            "backendPort" : 6443,
            "probe" : {
              "id" : "[concat(variables('masterLoadBalancerID'), '/probes/api-internal-probe')]"
            }
          }
        }
      ],
      "probes" : [
        {
          "name" : "api-internal-probe",
          "properties" : {
            "protocol" : "Https",
            "port" : 6443,
            "requestPath": "/readyz",
            "intervalInSeconds" : 10,
            "numberOfProbes" : 3
          }
        }
      ]
    }
  },
  {
    "apiVersion" : "2018-12-01",
    "type" : "Microsoft.Network/loadBalancers",
    "name" : "[variables('internalLoadBalancerName')]",
    "location" : "[variables('location')]",
    "sku": {
      "name": "[variables('skuName')]"
    },
    "properties" : {
      "frontendIPConfigurations" : [
        {
```

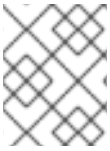
```
            "name" : "internal-lb-ip",
            "properties" : {
              "privateIPAllocationMethod" : "Dynamic",
              "subnet" : {
                "id" : "[variables('masterSubnetRef')]"
              },
              "privateIPAddressVersion" : "IPv4"
            }
          }
        ],
        "backendAddressPools" : [
          {
            "name" : "internal-lb-backend"
          }
        ],
        "loadBalancingRules" : [
          {
            "name" : "api-internal",
            "properties" : {
              "frontendIPConfiguration" : {
                "id" : "[concat(variables('internalLoadBalancerID'), '/frontendIPConfigurations/internal-lb-
ip')]"
              },
              "frontendPort" : 6443,
              "backendPort" : 6443,
              "enableFloatingIP" : false,
              "idleTimeoutInMinutes" : 30,
              "protocol" : "Tcp",
              "enableTcpReset" : false,
              "loadDistribution" : "Default",
              "backendAddressPool" : {
                "id" : "[concat(variables('internalLoadBalancerID'), '/backendAddressPools/internal-lb-
backend')]"
              },
              "probe" : {
                "id" : "[concat(variables('internalLoadBalancerID'), '/probes/api-internal-probe')]"
              }
            }
          },
          {
            "name" : "sint",
            "properties" : {
              "frontendIPConfiguration" : {
                "id" : "[concat(variables('internalLoadBalancerID'), '/frontendIPConfigurations/internal-lb-
ip')]"
              },
              "frontendPort" : 22623,
              "backendPort" : 22623,
              "enableFloatingIP" : false,
              "idleTimeoutInMinutes" : 30,
              "protocol" : "Tcp",
              "enableTcpReset" : false,
              "loadDistribution" : "Default",
              "backendAddressPool" : {
                "id" : "[concat(variables('internalLoadBalancerID'), '/backendAddressPools/internal-lb-
backend')]"
```

```
            },
            "probe" : {
              "id" : "[concat(variables('internalLoadBalancerID'), '/probes/sint-probe')]"
            }
          }
        }
      ],
      "probes" : [
        {
          "name" : "api-internal-probe",
          "properties" : {
            "protocol" : "Https",
            "port" : 6443,
            "requestPath": "/readyz",
            "intervalInSeconds" : 10,
            "numberOfProbes" : 3
          }
        },
        {
          "name" : "sint-probe",
          "properties" : {
            "protocol" : "Https",
            "port" : 22623,
            "requestPath": "/healthz",
            "intervalInSeconds" : 10,
            "numberOfProbes" : 3
          }
        }
      ]
    }
  },
  {
    "apiVersion": "2018-09-01",
    "type": "Microsoft.Network/privateDnsZones/A",
    "name": "[concat(parameters('privateDNSZoneName'), '/api')]",
    "location" : "[variables('location')]",
    "dependsOn" : [
      "[concat('Microsoft.Network/loadBalancers/', variables('internalLoadBalancerName'))]"
    ],
    "properties": {
      "ttl": 60,
      "aRecords": [
        {
          "ipv4Address": "
[reference(variables('internalLoadBalancerName')).frontendIPConfigurations[0].properties.privateIP
Address]"
        }
      ]
    }
  },
  {
    "apiVersion": "2018-09-01",
    "type": "Microsoft.Network/privateDnsZones/A",
    "name": "[concat(parameters('privateDNSZoneName'), '/api-int')]",
    "location" : "[variables('location')]",
    "dependsOn" : [
```

```
        "[concat('Microsoft.Network/loadBalancers/', variables('internalLoadBalancerName'))]"
      ],
      "properties": {
        "ttl": 60,
        "aRecords": [
          {
            "ipv4Address": "
[reference(variables('internalLoadBalancerName')).frontendIPConfigurations[0].properties.privateIP
Address]"
          }
        ]
      }
    }
  ]
}
```

3.9.14. 在 Azure 中创建 bootstrap 机器

您必须在 Microsoft Azure 中创建 bootstrap 机器，以便在 OpenShift Container Platform 集群初始化过程中使用。创建此机器的一种方法是修改提供的 Azure Resource Manager（ARM）模板。



注意

如果不使用提供的 ARM 模板来创建 bootstrap 机器，您必须检查提供的信息并手动创建基础架构。如果集群没有正确初始化，您可能需要联系红帽支持并提供您的安装日志。

先决条件

- 配置 Azure 帐户。
- 为集群生成 Ignition 配置文件。
- 在 Azure 中创建和配置 VNet 及相关子网。
- 在 Azure 中创建和配置联网及负载均衡器。
- 创建 control plane 和计算角色。

流程

1. 复制 **bootstrap 机器的 ARM 模板**一节中的模板，并将它以 **04_bootstrap.json** 保存到集群的安装目录中。此模板描述了集群所需的 bootstrap 机器。
2. 导出 bootstrap 机器部署所需的以下变量：

```
$ export BOOTSTRAP_URL=`az storage blob url --account-name ${CLUSTER_NAME}sa --
account-key ${ACCOUNT_KEY} -c "files" -n "bootstrap.ign" -o tsv`
$ export BOOTSTRAP_IGNITION=`jq -rcnM --arg v "3.1.0" --arg url ${BOOTSTRAP_URL}
'{ignition:{version:$v,config:{replace:{source:$url}}}}' | base64 | tr -d '\n'`
```

3. 使用 **az** CLI 创建部署：

```
$ az deployment group create -g ${RESOURCE_GROUP} \
```

```
--template-file "<installation_directory>/04_bootstrap.json" \
--parameters bootstrapIgnition="${BOOTSTRAP_IGNITION}" \ 1
--parameters sshKeyData="${SSH_KEY}" \ 2
--parameters baseName="${INFRA_ID}" 3
```

1. bootstrap 集群的 bootstrap Ignition 内容。
2. 作为字符串的 SSH RSA 公钥文件。
3. 资源名称使用的基本名称 ; 这通常是集群的基础架构 ID。

3.9.14.1. bootstrap 机器的 ARM 模板

您可以使用以下 Azure Resource Manager（ARM）模板来部署 OpenShift Container Platform 集群所需的 bootstrap 机器：

例 3.4. 04_bootstrap.json ARM 模板

```
{
  "$schema" : "https://schema.management.azure.com/schemas/2015-01-01/deploymentTemplate.json#",
  "contentVersion" : "1.0.0.0",
  "parameters" : {
    "baseName" : {
      "type" : "string",
      "minLength" : 1,
      "metadata" : {
        "description" : "Base name to be used in resource names (usually the cluster's Infra ID)"
      }
    },
    "bootstrapIgnition" : {
      "type" : "string",
      "minLength" : 1,
      "metadata" : {
        "description" : "Bootstrap ignition content for the bootstrap cluster"
      }
    },
    "sshKeyData" : {
      "type" : "securestring",
      "metadata" : {
        "description" : "SSH RSA public key file as a string."
      }
    },
    "bootstrapVMSize" : {
      "type" : "string",
      "defaultValue" : "Standard_D4s_v3",
      "allowedValues" : [
        "Standard_A2",
        "Standard_A3",
        "Standard_A4",
        "Standard_A5",
        "Standard_A6",
        "Standard_A7",
        "Standard_A8",
```

```
        "Standard_A9",
        "Standard_A10",
        "Standard_A11",
        "Standard_D2",
        "Standard_D3",
        "Standard_D4",
        "Standard_D11",
        "Standard_D12",
        "Standard_D13",
        "Standard_D14",
        "Standard_D2_v2",
        "Standard_D3_v2",
        "Standard_D4_v2",
        "Standard_D5_v2",
        "Standard_D8_v3",
        "Standard_D11_v2",
        "Standard_D12_v2",
        "Standard_D13_v2",
        "Standard_D14_v2",
        "Standard_E2_v3",
        "Standard_E4_v3",
        "Standard_E8_v3",
        "Standard_E16_v3",
        "Standard_E32_v3",
        "Standard_E64_v3",
        "Standard_E2s_v3",
        "Standard_E4s_v3",
        "Standard_E8s_v3",
        "Standard_E16s_v3",
        "Standard_E32s_v3",
        "Standard_E64s_v3",
        "Standard_G1",
        "Standard_G2",
        "Standard_G3",
        "Standard_G4",
        "Standard_G5",
        "Standard_DS2",
        "Standard_DS3",
        "Standard_DS4",
        "Standard_DS11",
        "Standard_DS12",
        "Standard_DS13",
        "Standard_DS14",
        "Standard_DS2_v2",
        "Standard_DS3_v2",
        "Standard_DS4_v2",
        "Standard_DS5_v2",
        "Standard_DS11_v2",
        "Standard_DS12_v2",
        "Standard_DS13_v2",
        "Standard_DS14_v2",
        "Standard_GS1",
        "Standard_GS2",
        "Standard_GS3",
        "Standard_GS4",
        "Standard_GS5",
```

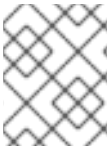
```
        "Standard_D2s_v3",
        "Standard_D4s_v3",
        "Standard_D8s_v3"
      ],
      "metadata" : {
        "description" : "The size of the Bootstrap Virtual Machine"
      }
    }
  },
  "variables" : {
    "location" : "[resourceGroup().location]",
    "virtualNetworkName" : "[concat(parameters('baseName'), '-vnet')]",
    "virtualNetworkID" : "[resourceId('Microsoft.Network/virtualNetworks',
variables('virtualNetworkName'))]",
    "masterSubnetName" : "[concat(parameters('baseName'), '-master-subnet')]",
    "masterSubnetRef" : "[concat(variables('virtualNetworkID'), '/subnets/',
variables('masterSubnetName'))]",
    "masterLoadBalancerName" : "[concat(parameters('baseName'), '-public-lb')]",
    "internalLoadBalancerName" : "[concat(parameters('baseName'), '-internal-lb')]",
    "sshKeyPath" : "/home/core/.ssh/authorized_keys",
    "identityName" : "[concat(parameters('baseName'), '-identity')]",
    "vmName" : "[concat(parameters('baseName'), '-bootstrap')]",
    "nicName" : "[concat(variables('vmName'), '-nic')]",
    "imageName" : "[concat(parameters('baseName'), '-image')]",
    "clusterNsgName" : "[concat(parameters('baseName'), '-nsg')]",
    "sshPublicIpAddressName" : "[concat(variables('vmName'), '-ssh-pip')]"
  },
  "resources" : [
    {
      "apiVersion" : "2018-12-01",
      "type" : "Microsoft.Network/publicIPAddresses",
      "name" : "[variables('sshPublicIpAddressName')]",
      "location" : "[variables('location')]",
      "sku": {
        "name": "Standard"
      },
      "properties" : {
        "publicIPAllocationMethod" : "Static",
        "dnsSettings" : {
          "domainNameLabel" : "[variables('sshPublicIpAddressName')]"
        }
      }
    },
    {
      "apiVersion" : "2018-06-01",
      "type" : "Microsoft.Network/networkInterfaces",
      "name" : "[variables('nicName')]",
      "location" : "[variables('location')]",
      "dependsOn" : [
        "[resourceId('Microsoft.Network/publicIPAddresses', variables('sshPublicIpAddressName'))]"
      ],
      "properties" : {
        "ipConfigurations" : [
          {
            "name" : "pipConfig",
            "properties" : {
```

```
            "privateIPAllocationMethod" : "Dynamic",
            "publicIPAddress": {
              "id": "[resourceId('Microsoft.Network/publicIPAddresses',
variables('sshPublicIpAddressName'))]"
            },
            "subnet" : {
              "id" : "[variables('masterSubnetRef')]"
            },
            "loadBalancerBackendAddressPools" : [
              {
                "id" : "[concat('/subscriptions/', subscription().subscriptionId, '/resourceGroups/',
resourceGroup().name, '/providers/Microsoft.Network/loadBalancers/',
variables('masterLoadBalancerName'), '/backendAddressPools/public-lb-backend')]"
              },
              {
                "id" : "[concat('/subscriptions/', subscription().subscriptionId, '/resourceGroups/',
resourceGroup().name, '/providers/Microsoft.Network/loadBalancers/',
variables('internalLoadBalancerName'), '/backendAddressPools/internal-lb-backend')]"
              }
            ]
          }
        }
      ]
    }
  },
  {
    "apiVersion" : "2018-06-01",
    "type" : "Microsoft.Compute/virtualMachines",
    "name" : "[variables('vmName')]",
    "location" : "[variables('location')]",
    "identity" : {
      "type" : "userAssigned",
      "userAssignedIdentities" : {
        "[resourceID('Microsoft.ManagedIdentity/userAssignedIdentities/',
variables('identityName'))]" : {}
      }
    },
    "dependsOn" : [
      "[concat('Microsoft.Network/networkInterfaces/', variables('nicName'))]"
    ],
    "properties" : {
      "hardwareProfile" : {
        "vmSize" : "[parameters('bootstrapVMSize')]"
      },
      "osProfile" : {
        "computerName" : "[variables('vmName')]",
        "adminUsername" : "core",
        "customData" : "[parameters('bootstrapIgnition')]",
        "linuxConfiguration" : {
          "disablePasswordAuthentication" : true,
          "ssh" : {
            "publicKeys" : [
              {
                "path" : "[variables('sshKeyPath')]",
                "keyData" : "[parameters('sshKeyData')]"
              }
```

```
            ]
          }
        }
      },
      "storageProfile" : {
        "imageReference": {
          "id": "[resourceId('Microsoft.Compute/images', variables('imageName'))]"
        },
        "osDisk" : {
          "name": "[concat(variables('vmName'),'_OSDisk')]",
          "osType" : "Linux",
          "createOption" : "FromImage",
          "managedDisk": {
            "storageAccountType": "Premium_LRS"
          },
          "diskSizeGB" : 100
        }
      },
      "networkProfile" : {
        "networkInterfaces" : [
          {
            "id" : "[resourceId('Microsoft.Network/networkInterfaces', variables('nicName'))]"
          }
        ]
      }
    },
    {
      "apiVersion" : "2018-06-01",
      "type": "Microsoft.Network/networkSecurityGroups/securityRules",
      "name" : "[concat(variables('clusterNsgName'), '/bootstrap_ssh_in')]",
      "location" : "[variables('location')]",
      "dependsOn" : [
        "[resourceId('Microsoft.Compute/virtualMachines', variables('vmName'))]"
      ],
      "properties": {
        "protocol" : "Tcp",
        "sourcePortRange" : "*",
        "destinationPortRange" : "22",
        "sourceAddressPrefix" : "*",
        "destinationAddressPrefix" : "*",
        "access" : "Allow",
        "priority" : 100,
        "direction" : "Inbound"
      }
    }
  ]
}
```

3.9.15. 在 Azure 中创建 control plane 机器

您必须在 Microsoft Azure 中创建 control plane 机器，供您的集群使用。创建这些机器的一种方法是修改提供的 Azure Resource Manager（ARM）模板。



注意

如果不使用提供的 ARM 模板来创建 control plane 机器，您必须检查提供的信息并手动创建基础架构。如果集群没有正确初始化，您可能需要联系红帽支持并提供您的安装日志。

先决条件

- 配置 Azure 帐户。
- 为集群生成 Ignition 配置文件。
- 在 Azure 中创建和配置 VNet 及相关子网。
- 在 Azure 中创建和配置联网及负载均衡器。
- 创建 control plane 和计算角色。
- 创建 bootstrap 机器。

流程

1. 复制 **control plane 机器的 ARM 模板**一节中的模板，并将它以 **05_masters.json** 保存到集群的安装目录中。此模板描述了集群所需的 control plane 机器。
2. 导出 control plane 机器部署所需的以下变量：

```
$ export MASTER_IGNITION=`cat <installation_directory>/master.ign | base64 | tr -d '\n'`
```

3. 使用 **az** CLI 创建部署：

```
$ az deployment group create -g ${RESOURCE_GROUP} \
  --template-file "<installation_directory>/05_masters.json" \
  --parameters masterIgnition="${MASTER_IGNITION}" \ 1
  --parameters sshKeyData="${SSH_KEY}" \ 2
  --parameters privateDNSZoneName="${CLUSTER_NAME}.${BASE_DOMAIN}" \ 3
  --parameters baseName="${INFRA_ID}" 4
```

1. control plane 节点的 Ignition 内容（也称为 master 节点）。
2. 作为字符串的 SSH RSA 公钥文件。
3. control plane 节点附加的私有 DNS 区域名称。
4. 资源名称使用的基本名称；这通常是集群的基础架构 ID。

3.9.15.1. control plane 机器的 ARM 模板

您可以使用以下 Azure Resource Manager（ARM）模板来部署 OpenShift Container Platform 集群所需的 control plane 机器：

例 3.5. 05_masters.json ARM 模板

```
{
  "$schema" : "https://schema.management.azure.com/schemas/2015-01-
```

```
01/deploymentTemplate.json#",
  "contentVersion" : "1.0.0.0",
  "parameters" : {
    "baseName" : {
      "type" : "string",
      "minLength" : 1,
      "metadata" : {
        "description" : "Base name to be used in resource names (usually the cluster's Infra ID)"
      }
    },
    "masterIgnition" : {
      "type" : "string",
      "metadata" : {
        "description" : "Ignition content for the master nodes"
      }
    },
    "numberOfMasters" : {
      "type" : "int",
      "defaultValue" : 3,
      "minValue" : 2,
      "maxValue" : 30,
      "metadata" : {
        "description" : "Number of OpenShift masters to deploy"
      }
    },
    "sshKeyData" : {
      "type" : "securestring",
      "metadata" : {
        "description" : "SSH RSA public key file as a string"
      }
    },
    "privateDNSZoneName" : {
      "type" : "string",
      "metadata" : {
        "description" : "Name of the private DNS zone the master nodes are going to be attached to"
      }
    },
    "masterVMSize" : {
      "type" : "string",
      "defaultValue" : "Standard_D8s_v3",
      "allowedValues" : [
        "Standard_A2",
        "Standard_A3",
        "Standard_A4",
        "Standard_A5",
        "Standard_A6",
        "Standard_A7",
        "Standard_A8",
        "Standard_A9",
        "Standard_A10",
        "Standard_A11",
        "Standard_D2",
        "Standard_D3",
        "Standard_D4",
        "Standard_D11",
        "Standard_D12",
```

```
        "Standard_D13",
        "Standard_D14",
        "Standard_D2_v2",
        "Standard_D3_v2",
        "Standard_D4_v2",
        "Standard_D5_v2",
        "Standard_D8_v3",
        "Standard_D11_v2",
        "Standard_D12_v2",
        "Standard_D13_v2",
        "Standard_D14_v2",
        "Standard_E2_v3",
        "Standard_E4_v3",
        "Standard_E8_v3",
        "Standard_E16_v3",
        "Standard_E32_v3",
        "Standard_E64_v3",
        "Standard_E2s_v3",
        "Standard_E4s_v3",
        "Standard_E8s_v3",
        "Standard_E16s_v3",
        "Standard_E32s_v3",
        "Standard_E64s_v3",
        "Standard_G1",
        "Standard_G2",
        "Standard_G3",
        "Standard_G4",
        "Standard_G5",
        "Standard_DS2",
        "Standard_DS3",
        "Standard_DS4",
        "Standard_DS11",
        "Standard_DS12",
        "Standard_DS13",
        "Standard_DS14",
        "Standard_DS2_v2",
        "Standard_DS3_v2",
        "Standard_DS4_v2",
        "Standard_DS5_v2",
        "Standard_DS11_v2",
        "Standard_DS12_v2",
        "Standard_DS13_v2",
        "Standard_DS14_v2",
        "Standard_GS1",
        "Standard_GS2",
        "Standard_GS3",
        "Standard_GS4",
        "Standard_GS5",
        "Standard_D2s_v3",
        "Standard_D4s_v3",
        "Standard_D8s_v3"
      ],
      "metadata" : {
        "description" : "The size of the Master Virtual Machines"
      }
    },
```

```
    "diskSizeGB" : {
      "type" : "int",
      "defaultValue" : 1024,
      "metadata" : {
        "description" : "Size of the Master VM OS disk, in GB"
      }
    }
  },
  "variables" : {
    "location" : "[resourceGroup().location]",
    "virtualNetworkName" : "[concat(parameters('baseName'), '-vnet')]",
    "virtualNetworkID" : "[resourceId('Microsoft.Network/virtualNetworks',
variables('virtualNetworkName'))]",
    "masterSubnetName" : "[concat(parameters('baseName'), '-master-subnet')]",
    "masterSubnetRef" : "[concat(variables('virtualNetworkID'), '/subnets/',
variables('masterSubnetName'))]",
    "masterLoadBalancerName" : "[concat(parameters('baseName'), '-public-lb')]",
    "internalLoadBalancerName" : "[concat(parameters('baseName'), '-internal-lb')]",
    "sshKeyPath" : "/home/core/.ssh/authorized_keys",
    "identityName" : "[concat(parameters('baseName'), '-identity')]",
    "imageName" : "[concat(parameters('baseName'), '-image')]",
    "copy" : [
      {
        "name" : "vmNames",
        "count" :  "[parameters('numberOfMasters')]",
        "input" : "[concat(parameters('baseName'), '-master-', copyIndex('vmNames'))]"
      }
    ]
  },
  "resources" : [
    {
      "apiVersion" : "2018-06-01",
      "type" : "Microsoft.Network/networkInterfaces",
      "copy" : {
        "name" : "nicCopy",
        "count" : "[length(variables('vmNames'))]"
      },
      "name" : "[concat(variables('vmNames')[copyIndex()], '-nic')]",
      "location" : "[variables('location')]",
      "properties" : {
        "ipConfigurations" : [
          {
            "name" : "pipConfig",
            "properties" : {
              "privateIPAllocationMethod" : "Dynamic",
              "subnet" : {
                "id" : "[variables('masterSubnetRef')]"
              },
              "loadBalancerBackendAddressPools" : [
                {
                  "id" : "[concat('/subscriptions/', subscription().subscriptionId, '/resourceGroups/',
resourceGroup().name, '/providers/Microsoft.Network/loadBalancers/',
variables('masterLoadBalancerName'), '/backendAddressPools/public-lb-backend')]"
                },
                {
                  "id" : "[concat('/subscriptions/', subscription().subscriptionId, '/resourceGroups/',
```

```
resourceGroup().name, '/providers/Microsoft.Network/loadBalancers/',
variables('internalLoadBalancerName'), '/backendAddressPools/internal-lb-backend')]"
              }
            ]
          }
        }
      ]
    }
  },
  {
    "apiVersion": "2018-09-01",
    "type": "Microsoft.Network/privateDnsZones/SRV",
    "name": "[concat(parameters('privateDNSZoneName'), '/_etcd-server-ssl._tcp')]",
    "location" : "[variables('location')]",
    "properties": {
      "ttl": 60,
      "copy": [{
        "name": "srvRecords",
        "count": "[length(variables('vmNames'))]",
        "input": {
          "priority": 0,
          "weight" : 10,
          "port" : 2380,
          "target" : "[concat('etcd-', copyIndex('srvRecords'), '.',
parameters('privateDNSZoneName'))]"
        }
      }]
    }
  },
  {
    "apiVersion": "2018-09-01",
    "type": "Microsoft.Network/privateDnsZones/A",
    "copy" : {
      "name" : "dnsCopy",
      "count" : "[length(variables('vmNames'))]"
    },
    "name": "[concat(parameters('privateDNSZoneName'), '/etcd-', copyIndex())]",
    "location" : "[variables('location')]",
    "dependsOn" : [
      "[concat('Microsoft.Network/networkInterfaces/', concat(variables('vmNames')[copyIndex()], '-
nic'))]"
    ],
    "properties": {
      "ttl": 60,
      "aRecords": [
        {
          "ipv4Address": "[reference(concat(variables('vmNames')[copyIndex()], '-
nic')).ipConfigurations[0].properties.privateIPAddress]"
        }
      ]
    }
  },
  {
    "apiVersion" : "2018-06-01",
    "type" : "Microsoft.Compute/virtualMachines",
    "copy" : {
```

```
      "name" : "vmCopy",
      "count" : "[length(variables('vmNames'))]"
    },
    "name" : "[variables('vmNames')[copyIndex()]]",
    "location" : "[variables('location')]",
    "identity" : {
      "type" : "userAssigned",
      "userAssignedIdentities" : {
        "[resourceID('Microsoft.ManagedIdentity/userAssignedIdentities/',
variables('identityName'))]" : {}
      }
    },
    "dependsOn" : [
      "[concat('Microsoft.Network/networkInterfaces/', concat(variables('vmNames')[copyIndex()], '-
nic'))]",
      "[concat('Microsoft.Network/privateDnsZones/', parameters('privateDNSZoneName'),
'/A/etcd-', copyIndex())]",
      "[concat('Microsoft.Network/privateDnsZones/', parameters('privateDNSZoneName'),
'/SRV/_etcd-server-ssl._tcp')]"
    ],
    "properties" : {
      "hardwareProfile" : {
        "vmSize" : "[parameters('masterVMSize')]"
      },
      "osProfile" : {
        "computerName" : "[variables('vmNames')[copyIndex()]]",
        "adminUsername" : "core",
        "customData" : "[parameters('masterIgnition')]",
        "linuxConfiguration" : {
          "disablePasswordAuthentication" : true,
          "ssh" : {
            "publicKeys" : [
              {
                "path" : "[variables('sshKeyPath')]",
                "keyData" : "[parameters('sshKeyData')]"
              }
            ]
          }
        }
      },
      "storageProfile" : {
        "imageReference": {
          "id": "[resourceId('Microsoft.Compute/images', variables('imageName'))]"
        },
        "osDisk" : {
          "name": "[concat(variables('vmNames')[copyIndex()], '_OSDisk')]",
          "osType" : "Linux",
          "createOption" : "FromImage",
          "caching": "ReadOnly",
          "writeAcceleratorEnabled": false,
          "managedDisk": {
            "storageAccountType": "Premium_LRS"
          },
          "diskSizeGB" : "[parameters('diskSizeGB')]"
        }
      },
```

```
        "networkProfile" : {
          "networkInterfaces" : [
            {
              "id" : "[resourceId('Microsoft.Network/networkInterfaces', concat(variables('vmNames')
[copyIndex()], '-nic'))]",
              "properties": {
                "primary": false
              }
            }
          ]
        }
      }
    }
  ]
}
```

3.9.16. 等待 bootstrap 完成并删除 Azure 中的 bootstrap 资源

在 Microsoft Azure 中创建所有所需的基础架构后，请等待您通过安装程序生成的 Ignition 配置文件所置备的机器上完成 bootstrap 过程。

先决条件

- 配置 Azure 帐户。
- 为集群生成 Ignition 配置文件。
- 在 Azure 中创建和配置 VNet 及相关子网。
- 在 Azure 中创建和配置联网及负载均衡器。
- 创建 control plane 和计算角色。
- 创建 bootstrap 机器。
- 创建 control plane 机器。

流程

1. 更改到包含安装程序的目录，再运行以下命令：

```
$ ./openshift-install wait-for bootstrap-complete --dir <installation_directory> \ 1
    --log-level info 2
```

1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

2 要查看不同的安装详情，请指定 **warn**、**debug** 或 **error**，而不要指定 **info**。

如果命令退出且不显示 **FATAL** 警告，则代表您的 control plane 已被初始化。

2. 删除 bootstrap 资源：

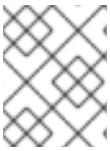
```
$ az network nsg rule delete -g ${RESOURCE_GROUP} --nsg-name ${INFRA_ID}-nsg --
```

```
name bootstrap_ssh_in
$ az vm stop -g ${RESOURCE_GROUP} --name ${INFRA_ID}-bootstrap
$ az vm deallocate -g ${RESOURCE_GROUP} --name ${INFRA_ID}-bootstrap
$ az vm delete -g ${RESOURCE_GROUP} --name ${INFRA_ID}-bootstrap --yes
$ az disk delete -g ${RESOURCE_GROUP} --name ${INFRA_ID}-bootstrap_OSDisk --no-
wait --yes
$ az network nic delete -g ${RESOURCE_GROUP} --name ${INFRA_ID}-bootstrap-nic --no-
wait
$ az storage blob delete --account-key ${ACCOUNT_KEY} --account-name
${CLUSTER_NAME}sa --container-name files --name bootstrap.ign
$ az network public-ip delete -g ${RESOURCE_GROUP} --name ${INFRA_ID}-bootstrap-
ssh-pip
```

3.9.17. 在 Azure 中创建额外的 worker 机器

您可以通过分散启动各个实例或利用集群外自动化流程（如自动缩放组），在 Microsoft Azure 中为您的集群创建 worker 机器。您还可以利用 OpenShift Container Platform 中的内置集群扩展机制和机器 API。

在本例中，您要使用 Azure Resource Manager（ARM） 模板来手动启动一个实例。通过在文件中添加类型为 **06_workers.json** 的其他资源，即可启动其他实例。



注意

如果不使用提供的 ARM 模板来创建 worker 机器，您必须检查提供的信息并手动创建基础架构。如果集群没有正确初始化，您可能需要联系红帽支持并提供您的安装日志。

先决条件

- 配置 Azure 帐户。
- 为集群生成 Ignition 配置文件。
- 在 Azure 中创建和配置 VNet 及相关子网。
- 在 Azure 中创建和配置联网及负载均衡器。
- 创建 control plane 和计算角色。
- 创建 bootstrap 机器。
- 创建 control plane 机器。

流程

1. 复制 **worker 机器的 ARM 模板**一节中的模板，并将它以 **06_workers.json** 保存到集群的安装目录中。此模板描述了集群所需的 worker 机器。
2. 导出 worker 机器部署所需的以下变量：

```
$ export WORKER_IGNITION=`cat <installation_directory>/worker.ign | base64 | tr -d '\n'`
```

3. 使用 **az** CLI 创建部署：

```
$ az deployment group create -g ${RESOURCE_GROUP} \
```

```
--template-file "<installation_directory>/06_workers.json" \
--parameters workerIgnition="${WORKER_IGNITION}" \ 1
--parameters sshKeyData="${SSH_KEY}" \ 2
--parameters baseName="${INFRA_ID}" 3
```

1 worker 节点的 Ignition 内容。

2 作为字符串的 SSH RSA 公钥文件。

3 资源名称使用的基本名称 ; 这通常是集群的基础架构 ID。

3.9.17.1. worker 机器的 ARM 模板

您可以使用以下 Azure Resource Manager（ARM）模板来部署 OpenShift Container Platform 集群所需的 worker 机器：

例 3.6. 06_workers.json ARM 模板

```
{
  "$schema" : "https://schema.management.azure.com/schemas/2015-01-01/deploymentTemplate.json#",
  "contentVersion" : "1.0.0.0",
  "parameters" : {
    "baseName" : {
      "type" : "string",
      "minLength" : 1,
      "metadata" : {
        "description" : "Base name to be used in resource names (usually the cluster's Infra ID)"
      }
    },
    "workerIgnition" : {
      "type" : "string",
      "metadata" : {
        "description" : "Ignition content for the worker nodes"
      }
    },
    "numberOfNodes" : {
      "type" : "int",
      "defaultValue" : 3,
      "minValue" : 2,
      "maxValue" : 30,
      "metadata" : {
        "description" : "Number of OpenShift compute nodes to deploy"
      }
    },
    "sshKeyData" : {
      "type" : "securestring",
      "metadata" : {
        "description" : "SSH RSA public key file as a string"
      }
    },
    "nodeVMSize" : {
      "type" : "string",
      "defaultValue" : "Standard_D4s_v3",
```

```
"allowedValues" : [
  "Standard_A2",
  "Standard_A3",
  "Standard_A4",
  "Standard_A5",
  "Standard_A6",
  "Standard_A7",
  "Standard_A8",
  "Standard_A9",
  "Standard_A10",
  "Standard_A11",
  "Standard_D2",
  "Standard_D3",
  "Standard_D4",
  "Standard_D11",
  "Standard_D12",
  "Standard_D13",
  "Standard_D14",
  "Standard_D2_v2",
  "Standard_D3_v2",
  "Standard_D4_v2",
  "Standard_D5_v2",
  "Standard_D8_v3",
  "Standard_D11_v2",
  "Standard_D12_v2",
  "Standard_D13_v2",
  "Standard_D14_v2",
  "Standard_E2_v3",
  "Standard_E4_v3",
  "Standard_E8_v3",
  "Standard_E16_v3",
  "Standard_E32_v3",
  "Standard_E64_v3",
  "Standard_E2s_v3",
  "Standard_E4s_v3",
  "Standard_E8s_v3",
  "Standard_E16s_v3",
  "Standard_E32s_v3",
  "Standard_E64s_v3",
  "Standard_G1",
  "Standard_G2",
  "Standard_G3",
  "Standard_G4",
  "Standard_G5",
  "Standard_DS2",
  "Standard_DS3",
  "Standard_DS4",
  "Standard_DS11",
  "Standard_DS12",
  "Standard_DS13",
  "Standard_DS14",
  "Standard_DS2_v2",
  "Standard_DS3_v2",
  "Standard_DS4_v2",
  "Standard_DS5_v2",
  "Standard_DS11_v2",
```

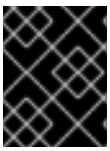
```
        "Standard_DS12_v2",
        "Standard_DS13_v2",
        "Standard_DS14_v2",
        "Standard_GS1",
        "Standard_GS2",
        "Standard_GS3",
        "Standard_GS4",
        "Standard_GS5",
        "Standard_D2s_v3",
        "Standard_D4s_v3",
        "Standard_D8s_v3"
      ],
      "metadata" : {
        "description" : "The size of the each Node Virtual Machine"
      }
    }
  },
  "variables" : {
    "location" : "[resourceGroup().location]",
    "virtualNetworkName" : "[concat(parameters('baseName'), '-vnet')]",
    "virtualNetworkID" : "[resourceId('Microsoft.Network/virtualNetworks',
variables('virtualNetworkName'))]",
    "nodeSubnetName" : "[concat(parameters('baseName'), '-worker-subnet')]",
    "nodeSubnetRef" : "[concat(variables('virtualNetworkID'), '/subnets/',
variables('nodeSubnetName'))]",
    "infraLoadBalancerName" : "[parameters('baseName')]",
    "sshKeyPath" : "/home/capi/.ssh/authorized_keys",
    "identityName" : "[concat(parameters('baseName'), '-identity')]",
    "imageName" : "[concat(parameters('baseName'), '-image')]",
    "copy" : [
      {
        "name" : "vmNames",
        "count" :  "[parameters('numberOfNodes')]",
        "input" : "[concat(parameters('baseName'), '-worker-', variables('location'), '-',
copyIndex('vmNames', 1))]"
      }
    ]
  },
  "resources" : [
    {
      "apiVersion" : "2019-05-01",
      "name" : "[concat('node', copyIndex())]",
      "type" : "Microsoft.Resources/deployments",
      "copy" : {
        "name" : "nodeCopy",
        "count" : "[length(variables('vmNames'))]"
      },
      "properties" : {
        "mode" : "Incremental",
        "template" : {
          "$schema" : "http://schema.management.azure.com/schemas/2015-01-
01/deploymentTemplate.json#",
          "contentVersion" : "1.0.0.0",
          "resources" : [
            {
              "apiVersion" : "2018-06-01",
```

```
            "type" : "Microsoft.Network/networkInterfaces",
            "name" : "[concat(variables('vmNames')[copyIndex()], '-nic')]",
            "location" : "[variables('location')]",
            "properties" : {
              "ipConfigurations" : [
                {
                  "name" : "pipConfig",
                  "properties" : {
                    "privateIPAllocationMethod" : "Dynamic",
                    "subnet" : {
                      "id" : "[variables('nodeSubnetRef')]"
                    }
                  }
                }
              ]
            }
          },
          {
            "apiVersion" : "2018-06-01",
            "type" : "Microsoft.Compute/virtualMachines",
            "name" : "[variables('vmNames')[copyIndex()]]",
            "location" : "[variables('location')]",
            "tags" : {
              "kubernetes.io-cluster-ffranzupi": "owned"
            },
            "identity" : {
              "type" : "userAssigned",
              "userAssignedIdentities" : {
                "[resourceID('Microsoft.ManagedIdentity/userAssignedIdentities/',
variables('identityName'))]" : {}
              }
            },
            "dependsOn" : [
              "[concat('Microsoft.Network/networkInterfaces/', concat(variables('vmNames')
[copyIndex()], '-nic'))]"
            ],
            "properties" : {
              "hardwareProfile" : {
                "vmSize" : "[parameters('nodeVMSize')]"
              },
              "osProfile" : {
                "computerName" : "[variables('vmNames')[copyIndex()]]",
                "adminUsername" : "capi",
                "customData" : "[parameters('workerIgnition')]",
                "linuxConfiguration" : {
                  "disablePasswordAuthentication" : true,
                  "ssh" : {
                    "publicKeys" : [
                      {
                        "path" : "[variables('sshKeyPath')]",
                        "keyData" : "[parameters('sshKeyData')]"
                      }
                    ]
                  }
                }
              },
```

```
            "storageProfile" : {
              "imageReference": {
                "id": "[resourceId('Microsoft.Compute/images', variables('imageName'))]"
              },
              "osDisk" : {
                "name": "[concat(variables('vmNames')[copyIndex()],'_OSDisk')]",
                "osType" : "Linux",
                "createOption" : "FromImage",
                "managedDisk": {
                  "storageAccountType": "Premium_LRS"
                },
                "diskSizeGB": 128
              }
            },
            "networkProfile" : {
              "networkInterfaces" : [
                {
                  "id" : "[resourceId('Microsoft.Network/networkInterfaces',
concat(variables('vmNames')[copyIndex()], '-nic'))]",
                  "properties": {
                    "primary": true
                  }
                }
              ]
            }
          }
        }
      ]
    }
  }
]
}
```

3.9.18. 通过下载二进制文件安装 OpenShift CLI

您需要安装 CLI（**oc**） 来使用命令行界面与 OpenShift Container Platform 进行交互。您可在 Linux、Windows 或 macOS 上安装 **oc**。



重要

如果安装了旧版本的 **oc**，则无法使用 OpenShift Container Platform 4.6 中的所有命令。下载并安装新版本的 **oc**。

3.9.18.1. 在 Linux 上安装 OpenShift CLI

您可以按照以下流程在 Linux 上安装 OpenShift CLI（**oc**）二进制文件。

流程

1. 进入到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 在 **Version** 下拉菜单中选择相应的版本。

3. 单击 **OpenShift v4.6 Linux 客户端** 条目旁边的 **Download Now**，再保存文件。
4. 解包存档：

```
$ tar xvzf <file>
```

5. 把 **oc** 二进制代码放到 **PATH** 中的目录中。
执行以下命令可以查看当前的 **PATH** 设置：

```
$ echo $PATH
```

安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
$ oc <command>
```

3.9.18.2. 在 Windows 上安装 OpenShift CLI

您可以按照以下流程在 Windows 上安装 OpenShift CLI（**oc**）二进制代码。

流程

1. 进入到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 在 **Version** 下拉菜单中选择相应的版本。
3. 单击 **OpenShift v4.6 Windows 客户端**条目旁边的 **Download Now**，再保存文件。
4. 使用 ZIP 程序解压存档。
5. 把 **oc** 二进制代码放到 **PATH** 中的目录中。
要查看您的 **PATH**，请打开命令提示窗口并执行以下命令：

```
C:\> path
```

安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
C:\> oc <command>
```

3.9.18.3. 在 macOS 上安装 OpenShift CLI

您可以按照以下流程在 macOS 上安装 OpenShift CLI（**oc**）二进制代码。

流程

1. 进入到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 在 **Version** 下拉菜单中选择相应的版本。
3. 单击 **OpenShift v4.6 MacOSX 客户端**条目旁边的 **Download Now**，再保存文件。
4. 解包和解压存档。
5. 将 **oc** 二进制文件移到 PATH 的目录中。

要查看您的 **PATH**，打开一个终端窗口并执行以下命令：

```
$ echo $PATH
```

安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
$ oc <command>
```

3.9.19. 使用 CLI 登录到集群

您可以通过导出集群 **kubeconfig** 文件，以默认系统用户身份登录集群。**kubeconfig** 文件包含关于集群的信息，供 CLI 用于将客户端连接到正确集群和 API 服务器。该文件特只适用于一个特定的集群，在 OpenShift Container Platform 安装过程中创建。

先决条件

- 已部署了 OpenShift Container Platform 集群。
- 已安装 **oc** CLI。

流程

1. 导出 **kubeadmin** 凭证：

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

2. 使用导出的配置，验证能否成功运行 **oc** 命令：

```
$ oc whoami
```

输出示例

```
system:admin
```

3.9.20. 批准机器的证书签名请求

将机器添加到集群时，会为您添加的每台机器生成两个待处理证书签名请求（CSR）。您必须确认这些 CSR 已获得批准，或根据需要自行批准。客户端请求必须首先被批准，然后是服务器请求。

先决条件

- 您已将机器添加到集群中。

流程

1. 确认集群可以识别这些机器：

```
$ oc get nodes
```

输出示例

```
NAME     STATUS   ROLES   AGE  VERSION
master-0 Ready    master  63m  v1.19.0
master-1 Ready    master  63m  v1.19.0
master-2 Ready    master  64m  v1.19.0
```

输出将列出您创建的所有机器。

注意

在一些 CSR 被批准前，以上输出可能不包括计算节点（也称为 worker 节点）。

2. 检查待处理的 CSR，并确保可以看到添加到集群中的每台机器都有 **Pending** 或 **Approved** 状态的客户端请求：

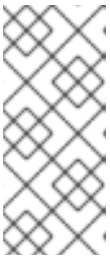
```
$ oc get csr
```

输出示例

```
NAME        AGE    REQUESTOR                                                    CONDITION
csr-8b2br   15m    system:serviceaccount:openshift-machine-config-operator:node-
bootstrapper   Pending
csr-8vnps   15m    system:serviceaccount:openshift-machine-config-operator:node-
bootstrapper   Pending
...
```

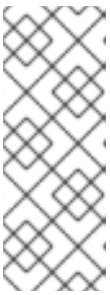
在本例中，两台机器加入了集群。您可能会在列表中看到更多已批准的 CSR。

3. 如果 CSR 没有获得批准，请在所添加机器的所有待处理 CSR 都处于 **Pending** 状态后，为您的集群机器批准这些 CSR：



注意

由于 CSR 会自动轮转，因此请在将机器添加到集群后一小时内批准您的 CSR。如果没有在一小时内批准，证书将会轮转，每个节点将会存在多个证书。您必须批准所有这些证书。批准客户端 CSR 后， Kubelet 为服务证书创建辅助 CSR，这需要手动批准。然后，如果 Kubelet 请求具有相同参数的新证书，则 **machine-approver** 会自动批准后续服务证书续订请求。



注意

对于在未启用机器 API 的平台中运行的集群，如裸机和其他用户置备的基础架构，必须采用一种方法自动批准 kubelet 提供证书请求（CSR）。如果没有批准请求，则 **oc exec**、**oc rsh** 和 **oc logs** 命令将无法成功，因为 API 服务器连接到 kubelet 时需要服务证书。与 Kubelet 端点联系的任何操作都需要此证书批准。这个方法必须监视新的 CSR，确认 CSR 由 **system:node** 或 **system:admin** 组中的 **node-bootstrapper** 服务帐户提交，并确认节点的身份。

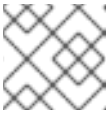
- 若要单独批准，请对每个有效的 CSR 运行以下命令：

```
$ oc adm certificate approve <csr_name> 1
```

1 **<csr_name>** 是当前 CSR 列表中 CSR 的名称。

- 要批准所有待处理的 CSR，请运行以下命令：

```
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}{{"\n"}}
{{end}}{{end}}' | xargs --no-run-if-empty oc adm certificate approve
```



注意

在有些 CSR 被批准前，一些 Operator 可能无法使用。

4. 现在，您的客户端请求已被批准，您必须查看添加到集群中的每台机器的服务器请求：

```
$ oc get csr
```

输出示例

```
NAME        AGE     REQUESTOR                                                   CONDITION
csr-bfd72   5m26s   system:node:ip-10-0-50-126.us-east-2.compute.internal
Pending
csr-c57lv   5m26s   system:node:ip-10-0-95-157.us-east-2.compute.internal
Pending
...
```

5. 如果剩余的 CSR 没有被批准，且处于 **Pending** 状态，请批准集群机器的 CSR：

- 若要单独批准，请对每个有效的 CSR 运行以下命令：

```
$ oc adm certificate approve <csr_name> 1
```

1 **<csr_name>** 是当前 CSR 列表中 CSR 的名称。

- 要批准所有待处理的 CSR，请运行以下命令：

```
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}{{"\n"}}
{{end}}{{end}}' | xargs oc adm certificate approve
```

6. 批准所有客户端和服务器 CSR 后，器将处于 **Ready** 状态。运行以下命令验证：

```
$ oc get nodes
```

输出示例

```
NAME      STATUS    ROLES   AGE  VERSION
master-0  Ready     master  73m  v1.20.0
master-1  Ready     master  73m  v1.20.0
master-2  Ready     master  74m  v1.20.0
worker-0  Ready     worker  11m  v1.20.0
worker-1  Ready     worker  11m  v1.20.0
```

注意

批准服务器 CSR 后可能需要几分钟时间让机器转换为 **Ready** 状态。

其他信息

- 如需有关 CSR 的更多信息，请参阅证书签名请求。

3.9.21. 添加 Ingress DNS 记录

如果您在创建 Kubernetes 清单并生成 Ignition 配置时删除了 DNS 区配置，您必须手动创建指向入口负载均衡器的 DNS 记录。您可以创建通配符 ***.apps.{baseDomain}.** 或具体的记录。您可以根据自己的要求使用 A、CNAME 和其他记录。

先决条件

- 已使用您置备的基础架构在 Microsoft Azure 上安装了 OpenShift Container Platform 集群。
- 安装 OpenShift CLI（**oc**）。
- 安装 **jq** 软件包。
- 安装或更新 Azure CLI。

流程

1. 确认 Ingress 路由器已创建了负载均衡器并填充 **EXTERNAL-IP** 字段：

```
$ oc -n openshift-ingress get service router-default
```

输出示例

```
NAME             TYPE           CLUSTER-IP      EXTERNAL-IP     PORT(S)                      AGE
router-default   LoadBalancer   172.30.20.10    35.130.120.110
80:32288/TCP,443:31215/TCP   20
```

2. 将 Ingress 路由器 IP 导出作为变量：

```
$ export PUBLIC_IP_ROUTER=`oc -n openshift-ingress get service router-default --no-
headers | awk '{print $4}'`
```

3. 在公共 DNS 区域中添加 ***.apps** 记录。

   a. 如果您要将此集群添加到新的公共区，请运行：

```
$ az network dns record-set a add-record -g ${BASE_DOMAIN_RESOURCE_GROUP} -
z ${CLUSTER_NAME}.${BASE_DOMAIN} -n *.apps -a ${PUBLIC_IP_ROUTER} --ttl 300
```

   b. 如果您要将此集群添加到已经存在的公共区中，请运行：

```
$ az network dns record-set a add-record -g ${BASE_DOMAIN_RESOURCE_GROUP} -
z ${BASE_DOMAIN} -n *.apps.${CLUSTER_NAME} -a ${PUBLIC_IP_ROUTER} --ttl 300
```

4. 在私有 DNS 区域中添加 ***.apps** 记录：

   a. 使用以下命令创建 ***.apps** 记录：

   ```
   $ az network private-dns record-set a create -g ${RESOURCE_GROUP} -z
   ${CLUSTER_NAME}.${BASE_DOMAIN} -n *.apps --ttl 300
   ```

   b. 使用以下命令在专用 DNS 区域中添加 ***.apps** 记录：

   ```
   $ az network private-dns record-set a add-record -g ${RESOURCE_GROUP} -z
   ${CLUSTER_NAME}.${BASE_DOMAIN} -n *.apps -a ${PUBLIC_IP_ROUTER}
   ```

如果需要添加特定域而不使用通配符，可以为集群的每个当前路由创建条目：

```
$ oc get --all-namespaces -o jsonpath='{range .items[*]}{range .status.ingress[*]}{.host}{"\n"}{end}
{end}' routes
```

输出示例

```
oauth-openshift.apps.cluster.basedomain.com
console-openshift-console.apps.cluster.basedomain.com
downloads-openshift-console.apps.cluster.basedomain.com
alertmanager-main-openshift-monitoring.apps.cluster.basedomain.com
grafana-openshift-monitoring.apps.cluster.basedomain.com
prometheus-k8s-openshift-monitoring.apps.cluster.basedomain.com
```

3.9.22. 在用户置备的基础架构上完成 Azure 安装

在 Microsoft Azure 用户置备的基础架构上启动 OpenShift Container Platform 安装后，您可以监控集群事件，直到集群就绪可用。

先决条件

- 在用户置备的 Azure 基础架构上为 OpenShift Container Platform 集群部署 bootstrap 机器。
- 安装 **oc** CLI 并登录。

流程

- 完成集群安装：

  ```
  $ ./openshift-install --dir <installation_directory> wait-for install-complete 1
  ```

  输出示例

  ```
  INFO Waiting up to 30m0s for the cluster to initialize...
  ```

  1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

重要

- 安装程序生成的 Ignition 配置文件包含在 24 小时后过期的证书，然后在过期时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外情况是，您需要手动批准待处理的 **node-bootstrapper** 证书签名请求（CSR）来恢复 kubelet 证书。如需更多信息，请参阅*从过期的 control plane 证书中恢复*的文档。
- 建议您在生成 12 小时后使用 Ignition 配置文件，因为集群安装后 24 小时证书从 16 小时轮转至 22 小时。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中运行证书更新时避免安装失败。

3.9.23. OpenShift Container Platform 的 Telemetry 访问

在 OpenShift Container Platform 4.6 中，默认运行的 Telemetry 服务提供有关集群健康状况和成功更新的指标，需要访问互联网。如果您的集群连接到互联网，Telemetry 会自动运行，而且集群会注册到 OpenShift Cluster Manager。

确认 OpenShift Cluster Manager 清单正确后，可以由 Telemetry 自动维护，也可以使用 OpenShift Cluster Manager 手动维护，使用订阅监控来跟踪帐户或多集群级别的 OpenShift Container Platform 订阅。

其他资源

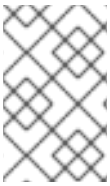
- 有关 Telemetry 服务的更多信息，请参阅关于远程健康监控。

3.10. 在 AZURE 上卸载集群

您可以删除部署到 Microsoft Azure 的集群。

3.10.1. 删除使用安装程序置备的基础架构的集群

您可以从云中删除使用安装程序置备的基础架构的集群。



注意

卸载后，检查云供应商是否有没有被正确移除的资源，特别是 User Provisioned Infrastructure（UPI）集群。可能存在安装程序没有创建的资源，或者安装程序无法访问的资源。

先决条件

- 有部署集群时所用的安装程序副本。
- 有创建集群时安装程序所生成的文件。

流程

1. 在用来安装集群的计算机中包含安装程序的目录中，运行以下命令：

```
$ ./openshift-install destroy cluster \
--dir <installation_directory> --log-level info 1 2
```

1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

2 要查看不同的详情，请指定 **warn**、**debug** 或 **error**，而不要指定 **info**。

注意

您必须为集群指定包含集群定义文件的目录。安装程序需要此目录中的 **metadata.json** 文件来删除集群。

2. 可选：删除 **<installation_directory>** 目录和 OpenShift Container Platform 安装程序。

第 4 章 在 GCP 上安装

4.1. 配置 GCP 项目

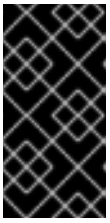
在安装 OpenShift Container Platform 之前，您必须配置 Google Cloud Platform (GCP) 项目来托管它。

4.1.1. 创建一个 GCP 项目

为了安装 OpenShift Container Platform，您必须在您的 Google Cloud Platform (GCP) 账户中创建一个项目来托管它。

流程

- 创建一个项目来托管您的 OpenShift Container Platform 集群。请参阅 GCP 文档中的创建和管理项目。



重要

如果您使用安装程序置备的基础架构，您的 GCP 项目必须使用 Premium Network Service Tier。使用安装程序安装的集群不支持 Standard Network Service Tier。安装程序为 **api-int.<cluster_name>.<base_domain>** URL 配置内部负载均衡；内部负载均衡需要 Premium Tier。

4.1.2. 在 GCP 中启用 API 服务

Google Cloud Platform (GCP) 项目需要访问一些 API 服务来完成 OpenShift Container Platform 安装。

先决条件

- 创建了用于托管集群的项目。

流程

- 在托管集群的项目中启用如下所需的 API 服务。请参阅 GCP 文档中的启用服务。

表 4.1. 所需的 API 服务

API 服务	控制台服务名称
compute Engine API	**compute.googleapis.com**
Google Cloud API	**cloudapis.googleapis.com**
Cloud Resource Manager API	**cloudresourcemanager.googleapis.com**
Google DNS API	**dns.googleapis.com**
IAM Service Account Credentials API	**iamcredentials.googleapis.com**

API 服务	控制台服务名称
Identity and Access Management (IAM) API	**iam.googleapis.com**
Service Management API	**servicemanagement.googleapis.com**
Service Usage API	**serviceusage.googleapis.com**
Google Cloud Storage JSON API	**storage-api.googleapis.com**
Cloud Storage	**storage-component.googleapis.com**

4.1.3. 为 GCP 配置 DNS

要安装 OpenShift Container Platform，您使用的 Google Cloud Platform (GCP) 帐户必须在托管 OpenShift Container Platform 集群的同一项目中有一个专用的公共托管区。此区域必须对域具有权威。DNS 服务为集群外部连接提供集群 DNS 解析和名称查询。

流程

1. 标识您的域或子域，以及注册商（registrar）。您可以转移现有的域和注册商，或通过 GCP 或其他来源获取新的域和注册商。

注意

如果您购买新域，则需要时间来传播相关的 DNS 更改。有关通过 Google 购买域的更多信息，请参阅 Google Domains。

2. 在 GCP 项目中为您的域或子域创建一个公共托管区。请参阅 GCP 文档中的创建公共区。使用合适的根域（如 **openshiftcorp.com**）或子域（如 **clusters.openshiftcorp.com**）。

3. 从托管区记录中提取新的权威名称服务器。请参阅 GCP 文档中的查找您的云 DNS 名称服务器。您通常有四个名称服务器。

4. 更新域所用名称服务器的注册商记录。例如，如果您将域注册到了 Google Domains，请参阅 Google Domains 帮助中的以下主题：如何切换到自定义名称服务器。

5. 如果您将根域迁移到 Google Cloud DNS，请迁移您的 DNS 记录。请参阅 GCP 文档中的 Migrating to Cloud DNS 。

6. 如果您使用子域，请按照您公司的流程将其委派记录添加到父域。这个过程可能包括对您公司的 IT 部门或控制您公司的根域和 DNS 服务的部门提出请求。

4.1.4. GCP 帐户限值

OpenShift Container Platform 集群使用诸多 Google Cloud Platform (GCP) 组件，默认的配额不会影响您安装默认 OpenShift Container Platform 集群的能力。

默认集群中包含三台计算机器和三台 control plane 机器，使用了以下资源。请注意，一些资源只在 bootstrap 过程中需要，会在集群部署后删除。

表 4.2. 默认集群中使用的 GCP 资源

服务	组件	位置	所需的资源总数	bootstrap 后删除的资源
服务帐户	IAM	全局	5	0
防火墙规则	Compute	全局	11	1
转发规则	Compute	全局	2	0
使用的全局 IP 地址	Compute	全局	4	1
健康检查	Compute	全局	3	0
镜像	Compute	全局	1	0
网络	Compute	全局	2	0
静态 IP 地址	Compute	区域	4	1
路由器	Compute	全局	1	0
Routes	Compute	全局	2	0
子网	Compute	全局	2	0
目标池	Compute	全局	3	0
CPU	Compute	区域	28	4
持久性磁盘 SSD (GB)	Compute	区域	896	128

注意

如果在安装过程中存在任何配额不足的情况，安装程序会显示一个错误信息，包括超过哪个配额，以及相关的区域。

请考虑您的集群的实际大小、预定的集群增长以及来自与您的帐户关联的其它集群的使用情况。CPU、静态 IP 地址和持久性磁盘 SSD (storage) 配额是最可能不足的。

如果您计划在以下区域之一部署集群，您将超过最大存储配额，并可能会超过 CPU 配额限制：

- **asia-east2**
- **asia-northeast2**
- **asia-south1**

- **domain-southeast1**
- **europe-north1**
- **europe-west2**
- **europe-west3**
- **europe-west6**
- **northamerica-northeast1**
- **southamerica-east1**
- **us-west2**

您可以从 GCP 控制台增加资源配额，但可能需要提交一个支持问题单。务必提前规划集群大小，以便在安装 OpenShift Container Platform 集群前有足够的时间来等待支持问题单被处理。

4.1.5. 在 GCP 中创建服务帐户

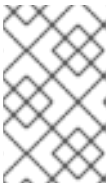
OpenShift Container Platform 需要一个 Google Cloud Platform (GCP) 服务帐户，用于访问 Google API 中数据的验证和授权。如果您没有包含项目中所需角色的现有 IAM 服务帐户，您必须创建一个。

先决条件

- 创建了用于托管集群的项目。

流程

1. 在用于托管 OpenShift Container Platform 集群的项目中，创建一个新服务帐户。请参阅 GCP 文档中的创建服务账户。
2. 为服务帐户授予适当的权限。您可以逐一授予权限，也可以为其分配 **Owner** 角色。请参阅将角色授予特定资源的服务帐户。



注意

获得所需权限最简单的方法是把服务帐户设置为项目的拥有者（owner），这意味着该服务帐户对项目有完全的控制权。您必须考虑这样设定所带来的的风险是否可以接受。

3. 以 JSON 格式创建服务帐户密钥。请参阅 GCP 文档中的创建服务帐户密钥。
创建集群时需要该服务帐户密钥。

4.1.5.1. 所需的 GCP 权限

如果将 **Owner** 角色附加到您创建的服务帐户，您向该服务帐户授予所有的权限，包括安装 OpenShift Container Platform 所需的权限。要部署 OpenShift Container Platform 集群，服务帐户需要以下权限：
如果您将集群部署到现有的 VPC 中，则服务帐户不需要某些网络权限，如下表所示：

安装程序所需的角色

- Compute Admin

- Security Admin
- Service Account Admin
- Service Account User
- Storage Admin

安装过程中创建网络资源所需的角色

- DNS Administrator

可选角色

若要使集群为其 Operator 创建新的有限凭证，请添加以下角色：

- Service Account Key Admin

以下角色将应用到 control plane 或计算机器使用的服务帐户：

表 4.3. GCP 服务帐户权限

帐户	角色
Control Plane	**roles/compute.instanceAdmin**
	roles/compute.networkAdmin
	roles/compute.securityAdmin
	roles/storage.admin
	roles/iam.serviceAccountUser
Compute	**roles/compute.viewer**
	roles/storage.admin

4.1.6. 支持的 GCP 区域

您可以将 OpenShift Container Platform 集群部署到下列 Google Cloud Platform (GCP) 区域：

- **asia-east1** (Changhua County, Taiwan)
- **asia-east2** (Hong Kong)
- **asia-northeast1** (Tokyo, Japan)
- **asia-northeast2** (Osaka, Japan)
- **asia-northeast3** (Seoul, South Korea)
- **asia-south1** (Mumbai, India)

- **asia-southeast1** (Jurong West, Singapore)
- **asia-southeast2** (Jakarta, India)
- **australia-southeast1** (Sydney, Australia)
- **europe-north1** (Hamina, Finland)
- **europe-west1** (St. Ghislain, Belgium)
- **europe-west2** (London, England, UK)
- **europe-west3** (Frankfurt, Germany)
- **europe-west4** (Eemshaven, Netherlands)
- **europe-west6** (Zürich, Switzerland)
- **northamerica-northeast1** (Montréal, Québec, Canada)
- **southamerica-east1** (São Paulo, Brazil)
- **us-central1** (Council Bluffs, Iowa, USA)
- **us-east1** (Moncks Corner, South Carolina, USA)
- **us-east4** (Ashburn, Northern Virginia, USA)
- **us-west1** (The Dalles, Oregon, USA)
- **us-west2** (Los Angeles, California, USA)
- **us-west3** (Salt Lake City, Utah, USA)
- **us-west4** (Las Vegas, Nevada, USA)

4.1.7. 后续步骤

- 在 GCP 上安装 OpenShift Container Platform 集群。您可以安装自定义集群，或使用默认选项快速安装集群。

4.2. 为 GCP 手动创建 IAM

在无法访问云身份和访问管理（IAM）API 的环境中，或者管理员更不希望将管理员级别的凭证 secret 存储在集群 **kube-system** 命名空间中时，可以在安装前将 Cloud Credential Operator（CCO）放入手动模式。

4.2.1. 在 kube-system 项目中存储管理员级别的 secret 的替代方案

Cloud Credential Operator（CCO） 将云供应商凭证作为 Kubernetes 自定义资源定义 (CRD) 进行管理。您可以通过在 **install-config.yaml** 文件中为 **credentialsMode** 参数设置不同的值，来配置 CCO 来满足机构的安全要求。

如果您不希望在集群 **kube-system** 项目中存储管理员级别的凭证 secret，您可以在安装 OpenShift Container Platform 时把 CCO 的 **credentialsMode** 参数设置为 **Manual**，并手动管理您的云凭证。

使用手动模式可允许每个集群组件只拥有所需的权限，而无需在集群中存储管理员级别的凭证。如果您的环境没有连接到云供应商公共 IAM 端点，您还可以使用此模式。但是，每次升级都必须手动将权限与新发行镜像协调。您还必须手动为每个请求它们的组件提供凭证。

其他资源

- 轮转或删除云供应商凭证。

有关所有可用 CCO 凭证模式及其支持的平台的更多信息，请参阅关于 Cloud Credential Operator。

4.2.2. 手动创建 IAM

在无法访问云身份和访问管理（IAM）API 的环境中，或者管理员更不希望将管理员级别的凭证 secret 存储在集群 **kube-system** 命名空间中时，可以在安装前将 Cloud Credential Operator（CCO）放入手动模式。

流程

1. 要生成清单，请在包含安装程序的目录中运行以下命令：

```
$ openshift-install create manifests --dir <installation_directory> 1
```

1 对于 **<installation_directory>**，请指定用于保存安装程序所创建的文件的目录名称。

2. 在 manifests 目录中插入配置映射，以便 Cloud Credential Operator 放置到手动模式中：

```
$ cat <<EOF > mycluster/manifests/cco-configmap.yaml
apiVersion: v1
kind: ConfigMap
metadata:
  name: cloud-credential-operator-config
  namespace: openshift-cloud-credential-operator
  annotations:
    release.openshift.io/create-only: "true"
data:
  disabled: "true"
EOF
```

3. 删除使用本地云凭证创建的 **admin** 凭证 secret。这会防止您的 **admin** 凭证存储在集群中：

```
$ rm mycluster/openshift/99_cloud-creds-secret.yaml
```

4. 从包含安装程序的目录中，获取 **openshift-install** 二进制文件要使用的 OpenShift Container Platform 发行镜像详情：

```
$ openshift-install version
```

输出示例

```
release image quay.io/openshift-release-dev/ocp-release:4.y.z-x86_64
```

5. 针对您要部署到的云，找到此发行版本镜像中的所有 **CredentialsRequests** 对象：

```
-
```

```
$ oc adm release extract quay.io/openshift-release-dev/ocp-release:4.y.z-x86_64 --
credentials-requests --cloud=gcp
```

这会显示每个请求的详情。

CredentialsRequest 对象示例

```
apiVersion: cloudcredential.openshift.io/v1
kind: CredentialsRequest
metadata:
  labels:
    controller-tools.k8s.io: "1.0"
  name: openshift-image-registry-gcs
  namespace: openshift-cloud-credential-operator
spec:
  secretRef:
    name: installer-cloud-credentials
    namespace: openshift-image-registry
  providerSpec:
    apiVersion: cloudcredential.openshift.io/v1
    kind: GCPProviderSpec
    predefinedRoles:
    - roles/storage.admin
    - roles/iam.serviceAccountUser
    skipServiceCheck: true
```

6. 在之前生成的 **openshift-install** 清单目录中为 secret 创建 YAML 文件。secret 必须使用在 **spec.secretRef** 中为每个 **credentialsRequest** 定义的命名空间和 secret 名称存储 。secret 数据的格式因云供应商而异。

7. 从包含安装程序的目录中，开始创建集群：

```
$ openshift-install create cluster --dir <installation_directory>
```

重要

在升级使用手动维护凭证的集群前，必须确保 CCO 处于可升级状态。详情请参阅您的云供应商的*对手动维护凭证的集群进行升级*部分的内容。

4.2.3. 管理凭证 root secret 格式

每个云供应商都使用 **kube-system** 命名空间中的一个凭证 root secret，用于满足所有凭证请求并创建它们相应的 secret。这可以通过 mint 新凭证（*mint mode*），或复制凭证 root secret（*passthrough mode*）实现。

secret 的格式因云而异，也用于每个 **CredentialsRequest** secret。

Google Cloud Platform（GCP）secret 格式

```
apiVersion: v1
kind: Secret
metadata:
  namespace: kube-system
```

```
  name: gcp-credentials
stringData:
  service_account.json: <ServiceAccount>
```

4.2.4. 使用手动维护的凭证升级集群

如果在未来的发行版本中添加了凭证，则使用手动维护凭证的集群的 Cloud Credential Operator（CCO）**可升级**状态会变为 **false**。对于次版本（例如从 4.5 到 4.6），这个状态会阻止升级，直到解决了更新的权限。对于 z-stream 版本（例如从 4.5.10 到 4.5.11），升级不会受阻，但必须为新版本更新凭证。

使用 Web 控制台的 **Administrator** 视角来判断 CCO 是否可以升级。

1. 导航至 **Administration → Cluster Settings**。
2. 要查看 CCO 状态详情，请点 **Cluster Operators** 列表中的 **cloud-credential**。
3. 如果 **Conditions** 部分中的 **Upgradeable** 状态为 **False**，请检查新发行版本的 **credentialsRequests**，并在升级前更新集群中手动维护的凭证以匹配。

除了为您要升级到的发行版本镜像创建新凭证外，还需要查看现有凭证所需的权限，并满足新发行版本中现有组件的所有新权限要求。CCO 无法检测到这些不匹配的问题，且在此情况下无法将 **upgradable** 设置为 **false**。

详情请参阅您的云供应商的*手动创建 IAM* 部分来了解如何获取和使用您的云所需的凭证。

4.2.5. Mint 模式

Mint 模式是 OpenShift Container Platform 的默认和推荐的 Cloud Credential Operator（CCO）凭证模式。在这种模式中，CCO 使用提供的管理员级云凭证来运行集群。AWS、GCP 和 Azure 支持 Mint 模式。

在 mint 模式中，**admin** 凭证存储在 **kube-system** 命名空间中，然后由 CCO 使用来处理集群中的 **CredentialsRequest** 对象，并为每个对象创建具有特定权限的用户。

mint 模式的好处包括：

- 每个集群组件只有其所需权限
- 云凭证的自动、持续协调，包括升级可能需要的额外凭证或权限

mint 模式的一个缺陷是，**admin** 凭证需要存储在集群 **kube-system** 的 secret 中。

4.2.6. 带有删除或轮转管理员凭证的 Mint 模式

目前，只有 AWS 支持这个模式。

在这个模式中，用户使用类似正常的 mint 模式的 **admin** 凭证安装 OpenShift Container Platform。但是，此模式会在集群安装后删除 **admin** 凭证 secret。

管理员可以让 Cloud Credential Operator 自行请求只读凭证，许它验证所有 **CredentialsRequest** 对象是否有其所需的权限。因此，除非需要更改内容，否则不需要 **admin** 凭证。删除关联的凭证后，可以根据需要在底层云上销毁它。

在升级前，应该恢复 **admin** 凭证。以后，如果凭证不存在，升级可能会阻止。

admin 凭证不会永久存储在集群中。

这个模式仍然需要在一个短的时间内，集群中存在 **admin** 凭证。它还需要为每个升级使用 **admin** 凭证手动重新生成 secret。

4.2.7. 后续步骤

- 安装 OpenShift Container Platform 集群：
  - 使用安装程序置备的基础架构默认选项在 GCP 上快速安装集群
  - 在安装程序置备的基础架构中使用云自定义安装集群
  - 使用网络自定义在安装程序置备的基础架构上安装集群

4.3. 在 GCP 上快速安装集群

在 OpenShift Container Platform 版本 4.6 中，您可以使用默认配置选项在 Google Cloud Platform（GCP）上安装集群。

4.3.1. 先决条件

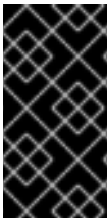
- 查看有关 OpenShift Container Platform 安装和更新流程的详细信息。
- 配置 GCP 帐户以托管集群。
- 如果使用防火墙，则必须将其配置为允许集群需要访问的站点。
- 如果不允许系统管理身份和访问管理（IAM），集群管理员可以 手动创建和维护 IAM 凭证。手动模式也可以用于云 IAM API 无法访问的环境中。

4.3.2. OpenShift Container Platform 的互联网访问

在 OpenShift Container Platform 4.6 中，您需要访问互联网来安装集群。

您必须具有以下互联网访问权限：

- 访问 OpenShift Cluster Manager 以下载安装程序并执行订阅管理。如果集群可以访问互联网，并且没有禁用 Telemetry，该服务会自动授权您的集群。
- 访问 Quay.io，以获取安装集群所需的软件包。
- 获取执行集群更新所需的软件包。



重要

如果您的集群无法直接访问互联网，则可以在置备的某些类基础架构上执行受限网络安装。在此过程中，您要下载所需的内容，并使用它在镜像 registry（mirror registry） 中填充安装集群并生成安装程序所需的软件包。对于某些安装类型，集群要安装到的环境不需要访问互联网。在更新集群之前，要更新 registry 镜像系统中的内容。

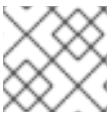
4.3.3. 生成 SSH 私钥并将其添加到代理中

如果要在集群上执行安装调试或灾难恢复，则必须为 **ssh-agent** 和安装程序提供 SSH 密钥。您可以使用此密钥访问公共集群中的 bootstrap 机器来排除安装问题。

注意

在生产环境中，您需要进行灾难恢复和调试。

您可以使用此密钥以 **core** 用户身份通过 SSH 连接到 master 节点。在部署集群时，此密钥会添加到 **core** 用户的 **~/.ssh/authorized_keys** 列表中。



注意

您必须使用一个本地密钥，而不要使用在特定平台上配置的密钥，如 AWS 密钥对。

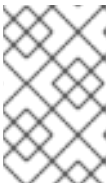
流程

1. 如果还没有为计算机上免密码身份验证而配置的 SSH 密钥，请创建一个。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ ssh-keygen -t ed25519 -N '' \
  -f <path>/<file_name> 1
```

1 指定新 SSH 密钥的路径和文件名，如 **~/.ssh/id_rsa**。如果您已有密钥对，请确保您的公钥位于 **~/.ssh** 目录中。

运行此命令会在指定的位置生成不需要密码的 SSH 密钥。



注意

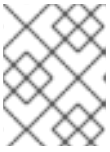
如果您计划在 **x86_64** 架构中安装使用 FIPS 验证的/Modules in Process 加密库的 OpenShift Container Platform 集群，不要创建使用 **ed25519** 算法的密钥。反之，创建一个使用 **rsa** 或 **ecdsa** 算法的密钥。

2. 作为后台任务启动 **ssh-agent** 进程：

```
$ eval "$(ssh-agent -s)"
```

输出示例

```
Agent pid 31874
```



注意

如果您的集群采用 FIPS 模式，则只使用 FIPS 兼容算法来生成 SSH 密钥。密钥必须是 RSA 或 ECDSA。

3. 将 SSH 私钥添加到 **ssh-agent**：

```
$ ssh-add <path>/<file_name> 1
```

输出示例

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

1 指定 SSH 私钥的路径和文件名，如 **~/.ssh/id_rsa**

4. 将 **GOOGLE_APPLICATION_CREDENTIALS** 环境变量设置为服务帐户私钥文件的完整路径。

```
$ export GOOGLE_APPLICATION_CREDENTIALS="<your_service_account_file>"
```

5. 验证是否应用了凭证。

```
$ gcloud auth list
```

后续步骤

- 在安装 OpenShift Container Platform 时，为安装程序提供 SSH 公钥。

4.3.4. 获取安装程序

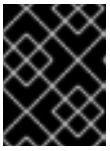
在安装 OpenShift Container Platform 之前，将安装文件下载到本地计算机上。

先决条件

- 运行 Linux 或 macOS 的计算机，本地磁盘空间为 500 MB

流程

1. 访问 OpenShift Cluster Manager 站点的 Infrastructure Provider 页面。如果您有红帽帐号，请使用自己的凭证登录。如果没有，请创建一个帐户。
2. 选择您的基础架构供应商。
3. 进入适用于您的安装类型的页面，下载您的操作系统的安装程序，并将文件放在要保存安装配置文件的目录中。。



重要

安装程序会在用来安装集群的计算机上创建若干文件。在完成集群安装后，您必须保留安装程序和安装程序所创建的文件。这两个文件都需要删除集群。

重要

删除安装程序创建的文件不会删除您的集群，即使集群在安装过程中失败也是如此。要删除集群，为特定云供应商完成 OpenShift Container Platform 卸载流程。

4. 提取安装程序。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ tar xvf openshift-install-linux.tar.gz
```

5. 从 Red Hat OpenShift Cluster Manager 下载安装 pull secret 。通过此 pull secret，您可以进行所含授权机构提供的服务的身份验证，这些服务包括为 OpenShift Container Platform 组件提供容器镜像的 Quay.io。

4.3.5. 部署集群

您可以在兼容云平台中安装 OpenShift Container Platform。

重要

安装程序的 **create cluster** 命令只能在初始安装过程中运行一次。

先决条件

- 配置托管集群的云平台的帐户。
- 获取 OpenShift Container Platform 安装程序以及集群的 pull secret。

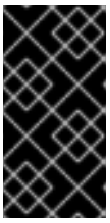
流程

1. 对于存储在以下位置的任何现有 GCP 凭证，删除不使用您为集群配置的 GCP 帐户的服务帐户密钥的凭证：
   - **GOOGLE_CREDENTIALS**、**GOOGLE_CLOUD_KEYFILE_JSON** 或 **GCLOUD_KEYFILE_JSON** 环境变量
   - **~/.gcp/osServiceAccount.json** 文件
   - **gcloud cli** 默认凭证
2. 更改为包含安装程序的目录并初始化集群部署：

```
$ ./openshift-install create cluster --dir <installation_directory> \ 1
    --log-level=info 2
```

1 对于 **<installation_directory>**，请指定用于保存安装程序所创建的文件的目录名称。

2 要查看不同的安装详情，请指定 **warn**、**debug** 或 **error**，而不要指定 **info**。



重要

指定一个空目录。一些安装信息，如 bootstrap X.509 证书，有较短的过期间隔，因此不要重复使用安装目录。如果要重复使用另一个集群安装中的个别文件，可以将其复制到您的目录中。但是，一些安装数据的文件名可能会在发行版本之间有所改变。从 OpenShift Container Platform 老版本中复制安装文件时要格外小心。

在提示符处提供值：

a. 可选：选择用来访问集群机器的 SSH 密钥。

注意

对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 **ssh-agent** 进程使用的 SSH 密钥。

b. 选择 **gcp** 作为目标平台。

c. 如果您没有为计算机上的 GCP 帐户配置服务帐户密钥，则必须从 GCP 获取，并粘贴文件的内容或输入文件的绝对路径。

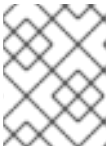
d. 选择要在其中置备集群的项目 ID。默认值由您配置的服务帐户指定。

e. 选择要在其中部署集群的区域。

f. 选择集群要部署到的基域。基域与您为集群创建的公共 DNS 区对应。

g. 为集群输入一个描述性名称。如果您提供的名称超过 6 个字符，只有前 6 个字符会用在从集群名称生成的基础架构 ID 中。

h. 粘贴 Red Hat OpenShift Cluster Manager 中的 pull secret 。



注意

如果您在主机上配置的云供应商帐户没有足够的权限来部署集群，安装过程将会停止，并且显示缺少的权限。

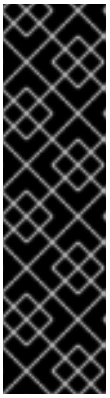
集群部署完成后，终端会显示访问集群的信息，包括指向其 Web 控制台的链接和 **kubeadmin** 用户的凭证。

输出示例

```
...
INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run 'export
KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'
INFO Access the OpenShift web-console here: https://console-openshift-
console.apps.mycluster.example.com
INFO Login to the console with user: "kubeadmin", and password: "4vYBz-Ee6gm-ymBZj-
Wt5AL"
INFO Time elapsed: 36m22s
```

注意

当安装成功时，集群访问和凭证信息还会输出到 **<installation_directory>/.openshift_install.log**。



重要

- 安装程序生成的 Ignition 配置文件包含在 24 小时后过期的证书，然后在过期时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外情况是，您需要手动批准待处理的 **node-bootstrapper** 证书签名请求（CSR）来恢复 kubelet 证书。如需更多信息，请参阅*从过期的 control plane 证书中恢复*的文档。

- 建议您在生成 12 小时后使用 Ignition 配置文件，因为集群安装后 24 小时证书从 16 小时轮转至 22 小时。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中运行证书更新时避免安装失败。

重要

您不得删除安装程序或安装程序所创建的文件。需要这两者才能删除集群。

3. 可选：您可以减少用于安装集群的服务帐户的权限数。

- 如果将 **Owner** 角色分配给您的服务帐户，您可以删除该角色并使用 **Viewer** 角色替换。
- 如果您包含 **Service Account Key Admin** 角色，您可以将其删除。

4.3.6. 通过下载二进制文件安装 OpenShift CLI

您需要安装 CLI（**oc**） 来使用命令行界面与 OpenShift Container Platform 进行交互。您可在 Linux 、Windows 或 macOS 上安装 **oc**。

重要

如果安装了旧版本的 **oc**，则无法使用 OpenShift Container Platform 4.6 中的所有命令。下载并安装新版本的 **oc**。

4.3.6.1. 在 Linux 上安装 OpenShift CLI

您可以按照以下流程在 Linux 上安装 OpenShift CLI（**oc**）二进制文件。

流程

1. 进入到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 在 **Version** 下拉菜单中选择相应的版本。
3. 单击 **OpenShift v4.6 Linux 客户端** 条目旁边的 **Download Now**，再保存文件。
4. 解包存档：

```
$ tar xvzf <file>
```

5. 把 **oc** 二进制代码放到 **PATH** 中的目录中。
执行以下命令可以查看当前的 **PATH** 设置：

```
$ echo $PATH
```

安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
$ oc <command>
```

4.3.6.2. 在 Windows 上安装 OpenShift CLI

您可以按照以下流程在 Windows 上安装 OpenShift CLI（**oc**）二进制代码。

流程

1. 进入到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 在 **Version** 下拉菜单中选择相应的版本。
3. 单击 **OpenShift v4.6 Windows 客户端**条目旁边的 **Download Now**，再保存文件。
4. 使用 ZIP 程序解压存档。

5. 把 **oc** 二进制代码放到 **PATH** 中的目录中。
   要查看您的 **PATH**，请打开命令提示窗口并执行以下命令：

```
C:\> path
```

安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
C:\> oc <command>
```

4.3.6.3. 在 macOS 上安装 OpenShift CLI

您可以按照以下流程在 macOS 上安装 OpenShift CLI（**oc**）二进制代码。

流程

1. 进入到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 在 **Version** 下拉菜单中选择相应的版本。
3. 单击 **OpenShift v4.6 MacOSX 客户端**条目旁边的 **Download Now**，再保存文件。
4. 解包和解压存档。
5. 将 **oc** 二进制文件移到 PATH 的目录中。
   要查看您的 **PATH**，打开一个终端窗口并执行以下命令：

```
$ echo $PATH
```

安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
$ oc <command>
```

4.3.7. 使用 CLI 登录到集群

您可以通过导出集群 **kubeconfig** 文件，以默认系统用户身份登录集群。**kubeconfig** 文件包含关于集群的信息，供 CLI 用于将客户端连接到正确集群和 API 服务器。该文件特只适用于一个特定的集群，在 OpenShift Container Platform 安装过程中创建。

先决条件

- 已部署了 OpenShift Container Platform 集群。
- 已安装 **oc** CLI。

流程

1. 导出 **kubeadmin** 凭证：

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

2. 使用导出的配置，验证能否成功运行 **oc** 命令：

```
$ oc whoami
```

输出示例

```
system:admin
```

其他资源

- 如需有关访问和了解 OpenShift Container Platform Web 控制台的更多信息，请参阅访问 Web 控制台。

4.3.8. OpenShift Container Platform 的 Telemetry 访问

在 OpenShift Container Platform 4.6 中，默认运行的 Telemetry 服务提供有关集群健康状况和成功更新的指标，需要访问互联网。如果您的集群连接到互联网，Telemetry 会自动运行，而且集群会注册到 OpenShift Cluster Manager。

确认 OpenShift Cluster Manager 清单正确后，可以由 Telemetry 自动维护，也可以使用 OpenShift Cluster Manager 手动维护，使用订阅监控来跟踪帐户或多集群级别的 OpenShift Container Platform 订阅。

其他资源

- 有关 Telemetry 服务的更多信息，请参阅关于远程健康监控。

4.3.9. 后续步骤

- 自定义集群。
- 若有需要，您可以选择不使用远程健康报告。

4.4. 使用自定义在 GCP 上安装集群

在 OpenShift Container Platform 版本 4.6 中，您可以在安装程序在 Google Cloud Platform（GCP）上置备的基础架构上安装自定义的集群。要自定义安装，请在安装集群前修改 **install-config.yaml** 文件中的参数。

4.4.1. 先决条件

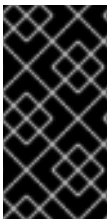
- 查看有关 OpenShift Container Platform 安装和更新流程的详细信息。
- 配置 GCP 帐户以托管集群。
- 如果使用防火墙，则必须将其配置为允许集群需要访问的站点。
- 如果不允许系统管理身份和访问管理（IAM），集群管理员可以 手动创建和维护 IAM 凭证。手动模式也可以用于云 IAM API 无法访问的环境中。

4.4.2. OpenShift Container Platform 的互联网访问

在 OpenShift Container Platform 4.6 中，您需要访问互联网来安装集群。

您必须具有以下互联网访问权限：

- 访问 OpenShift Cluster Manager 以下载安装程序并执行订阅管理。如果集群可以访问互联网，并且没有禁用 Telemetry，该服务会自动授权您的集群。
- 访问 Quay.io，以获取安装集群所需的软件包。
- 获取执行集群更新所需的软件包。

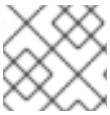


重要

如果您的集群无法直接访问互联网，则可以在置备的某些类基础架构上执行受限网络安装。在此过程中，您要下载所需的内容，并使用它在镜像 registry（mirror registry） 中填充安装集群并生成安装程序所需的软件包。对于某些安装类型，集群要安装到的环境不需要访问互联网。在更新集群之前，要更新 registry 镜像系统中的内容。

4.4.3. 生成 SSH 私钥并将其添加到代理中

如果要在集群上执行安装调试或灾难恢复，则必须为 **ssh-agent** 和安装程序提供 SSH 密钥。您可以使用此密钥访问公共集群中的 bootstrap 机器来排除安装问题。



注意

在生产环境中，您需要进行灾难恢复和调试。

您可以使用此密钥以 **core** 用户身份通过 SSH 连接到 master 节点。在部署集群时，此密钥会添加到 **core** 用户的 **~/.ssh/authorized_keys** 列表中。

注意

您必须使用一个本地密钥，而不要使用在特定平台上配置的密钥，如 AWS 密钥对。

流程

1. 如果还没有为计算机上免密码身份验证而配置的 SSH 密钥，请创建一个。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ ssh-keygen -t ed25519 -N '' \
  -f <path>/<file_name> 1
```

1 指定新 SSH 密钥的路径和文件名，如 **~/.ssh/id_rsa**。如果您已有密钥对，请确保您的公钥位于 **~/.ssh** 目录中。

运行此命令会在指定的位置生成不需要密码的 SSH 密钥。

注意

如果您计划在 **x86_64** 架构中安装使用 FIPS 验证的/Modules in Process 加密库的 OpenShift Container Platform 集群，不要创建使用 **ed25519** 算法的密钥。反之，创建一个使用 **rsa** 或 **ecdsa** 算法的密钥。

2. 作为后台任务启动 **ssh-agent** 进程：

```
$ eval "$(ssh-agent -s)"
```

输出示例

```
Agent pid 31874
```

注意

如果您的集群采用 FIPS 模式，则只使用 FIPS 兼容算法来生成 SSH 密钥。密钥必须是 RSA 或 ECDSA。

3. 将 SSH 私钥添加到 **ssh-agent**：

```
$ ssh-add <path>/<file_name> 1
```

输出示例

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

1 指定 SSH 私钥的路径和文件名，如 **~/.ssh/id_rsa**

4. 将 **GOOGLE_APPLICATION_CREDENTIALS** 环境变量设置为服务帐户私钥文件的完整路径。

```
$ export GOOGLE_APPLICATION_CREDENTIALS="<your_service_account_file>"
```

5. 验证是否应用了凭证。

```
$ gcloud auth list
```

后续步骤

- 在安装 OpenShift Container Platform 时，为安装程序提供 SSH 公钥。

4.4.4. 获取安装程序

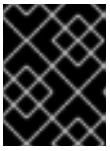
在安装 OpenShift Container Platform 之前，将安装文件下载到本地计算机上。

先决条件

- 运行 Linux 或 macOS 的计算机，本地磁盘空间为 500 MB

流程

1. 访问 OpenShift Cluster Manager 站点的 Infrastructure Provider 页面。如果您有红帽帐号，请使用自己的凭证登录。如果没有，请创建一个帐户。
2. 选择您的基础架构供应商。
3. 进入适用于您的安装类型的页面，下载您的操作系统的安装程序，并将文件放在要保存安装配置文件的目录中。。



重要

安装程序会在用来安装集群的计算机上创建若干文件。在完成集群安装后，您必须保留安装程序和安装程序所创建的文件。这两个文件都需要删除集群。

重要

删除安装程序创建的文件不会删除您的集群，即使集群在安装过程中失败也是如此。要删除集群，为特定云供应商完成 OpenShift Container Platform 卸载流程。

4. 提取安装程序。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ tar xvf openshift-install-linux.tar.gz
```

5. 从 Red Hat OpenShift Cluster Manager 下载安装 pull secret 。通过此 pull secret，您可以进行所含授权机构提供的服务的身份验证，这些服务包括为 OpenShift Container Platform 组件提供容器镜像的 Quay.io。

4.4.5. 创建安装配置文件

您可以自定义在 Google Cloud Platform (GCP) 上安装的 OpenShift Container Platform 集群。

先决条件

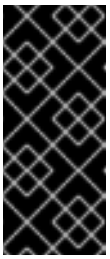
- 获取 OpenShift Container Platform 安装程序以及集群的 pull secret。

流程

1. 创建 **install-config.yaml** 文件。
   a. 更改到包含安装程序的目录，再运行以下命令：

```
$ ./openshift-install create install-config --dir <installation_directory> 1
```

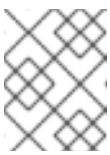
   1 对于 **<installation_directory>**，请指定用于保存安装程序所创建的文件的目录名称。



重要

指定一个空目录。一些安装信息，如 bootstrap X.509 证书，有较短的过期间隔，因此不要重复使用安装目录。如果要重复使用另一个集群安装中的个别文件，可以将其复制到您的目录中。但是，一些安装数据的文件名可能会在发行版本之间有所改变。从 OpenShift Container Platform 老版本中复制安装文件时要格外小心。

   b. 在提示符处，提供您的云的配置详情：
      i. 可选：选择用来访问集群机器的 SSH 密钥。



注意

对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 **ssh-agent** 进程使用的 SSH 密钥。

ii. 选择 **gcp** 作为目标平台。

iii. 如果您没有为计算机上的 GCP 帐户配置服务帐户密钥，则必须从 GCP 获取，并粘贴文件的内容或输入文件的绝对路径。

iv. 选择要在其中置备集群的项目 ID。默认值由您配置的服务帐户指定。

v. 选择要在其中部署集群的区域。

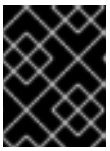
vi. 选择集群要部署到的基域。基域与您为集群创建的公共 DNS 区对应。

vii. 为集群输入一个描述性名称。

viii. 粘贴 Red Hat OpenShift Cluster Manager 中的 pull secret 。

2. 修改 **install-config.yaml** 文件。您可以在**安装配置参数**部分中找到有关可用参数的更多信息。

3. 备份 **install-config.yaml** 文件，以便用于安装多个集群。



重要

install-config.yaml 文件会在安装过程中消耗掉。如果要重复使用此文件，必须现在备份。

4.4.5.1. 安装配置参数

在部署 OpenShift Container Platform 集群前，您可以提供参数值，以描述托管集群的云平台的帐户并选择性地自定义集群平台。在创建 **install-config.yaml** 安装配置文件时，您可以通过命令行来提供所需的参数的值。如果要自定义集群，可以修改 **install-config.yaml** 文件来提供关于平台的更多信息。

注意

安装之后，您无法修改 **install-config.yaml** 文件中的这些参数。

重要

openshift-install 命令不验证参数的字段名称。如果指定了不正确的名称，则不会创建相关的文件或对象，且不会报告错误。确保所有指定的参数的字段名称都正确。

4.4.5.1.1. 所需的配置参数

下表描述了所需的安装配置参数：

表 4.4. 所需的参数

参数	描述	值
apiVersion	**install-config.yaml** 内容的 API 版本。当前版本是 **v1**。安装程序还可能支持旧的 API 版本。	字符串

参数	描述	值
baseDomain	云供应商的基域。此基础域用于创建到 OpenShift Container Platform 集群组件的路由。集群的完整 DNS 名称是 **baseDomain** 和 **metadata.name** 参数值的组合，其格式为 **<metadata.name>.<baseDomain>**。	完全限定域名或子域名，如 **example.com**。
metadata	Kubernetes 资源 **ObjectMeta**，其中只消耗 **name** 参数。	对象
metadata.name	集群的名称。集群的 DNS 记录是 **{{.metadata.name}}.{{.baseDomain}}** 的子域。	小写字母,连字符(-)和句点(.)的字符串，如 **dev**。
platform	执行安装的具体平台配置：**aws**、**baremetal**、**azure**、**openstack**、**ovirt**、**vsphere**。有关 **platform.<platform>** 参数的额外信息，请参考下表来了解您的具体平台。	对象

pullSecret	从 Red Hat OpenShift Cluster Manager 获取 pull secret，验证从 Quay.io 等服务中下载 OpenShift Container Platform 组件的容器镜像。	{
		"auths":{
		"cloud.openshift.com":{
		"auth":"b3Blb=",
		"email":"you@example.com"
		},
		"quay.io":{
		"auth":"b3Blb=",
		"email":"you@example.com"
		}
		}
		}

4.4.5.1.2. 网络配置参数

您可以根据现有网络基础架构的要求自定义安装配置。例如，您可以扩展集群网络的 IP 地址块，或者提供不同于默认值的不同 IP 地址块。

只支持 IPv4 地址。

表 4.5. 网络参数

参数	描述	值
networking	集群网络的配置。	对象  **注意** 您不能在安装后修改 **networking** 对象指定的参数。
networking.networkType	要安装的集群网络供应商 Container Network Interface（CNI）插件。	**OpenShiftSDN** 或 **OVNKubernetes**。默认值为 **OpenShiftSDN**。
networking.clusterNetwork	pod 的 IP 地址块。 默认值为 **10.128.0.0/14**，主机前缀为 **/23**。 如果您指定多个 IP 地址块，则块不得互相重叠。	一个对象数组。例如： networking: clusterNetwork: - cidr: 10.128.0.0/14 hostPrefix: 23
networking.clusterNetwork.cidr	使用 **networking.clusterNetwork** 时需要此项。IP 地址块。 一个 IPv4 网络。	使用 CIDR 形式的 IP 地址块。IPv4 块的前缀长度介于 **0** 到 **32** 之间。
networking.clusterNetwork.hostPrefix	分配给每个单独节点的子网前缀长度。例如，如果 **hostPrefix** 设为 **23**，则每个节点从所给的 **cidr** 中分配一个 **/23** 子网。**hostPrefix** 值 **23** 提供 510（2^(32 - 23)- 2）个 pod IP 地址。	子网前缀。 默认值为 **23**。
networking.serviceNetwork	服务的 IP 地址块。默认值为 **172.30.0.0/16**。 OpenShift SDN 和 OVN-Kubernetes 网络供应商只支持服务网络的一个 IP 地址块。	CIDR 格式具有 IP 地址块的数组。例如： networking: serviceNetwork: - 172.30.0.0/16
networking.machineNetwork	机器的 IP 地址块。 如果您指定多个 IP 地址块，则块不得互相重叠。	一个对象数组。例如： networking: machineNetwork: - cidr: 10.0.0.0/16

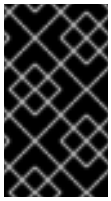
参数	描述	值
networking.machineNetwork.cidr	使用 **networking.machineNetwork** 时需要。IP 地址块。libvirt 以外的所有平台的默认值为 **10.0.0.0/16**。对于 libvirt，默认值为 **192.168.126.0/24**。	CIDR 表示法中的 IP 网络块。 例如： **10.0.0.0/16**。  **注意** 将 **networking.machineNetwork** 设置为与首选 NIC 所在的 CIDR 匹配。

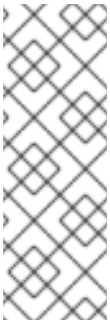
4.4.5.1.3. 可选配置参数

下表描述了可选安装配置参数：

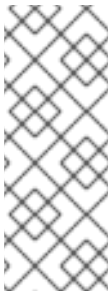
表 4.6. 可选参数

参数	描述	值
additionalTrustBundle	添加到节点可信证书存储中的 PEM 编码 X.509 证书捆绑包。配置了代理时，也可以使用这个信任捆绑包。	字符串
compute	组成计算节点的机器的配置。	machine-pool 对象的数组。详情请查看以下"Machine-pool"表。
compute.architecture	决定池中机器的指令集合架构。目前不支持异构集群，因此所有池都必须指定相同的架构。有效值为 **amd64** （默认值）。	字符串
compute.hyperthreading	是否在计算机器上启用或禁用并发多线程或**超线程**。默认情况下，启用并发多线程以提高机器内核的性能。  **重要** 如果禁用并发多线程，请确保在容量规划时考虑到机器性能可能会显著降低的问题。	**Enabled** 或 **Disabled**
compute.name	使用 **compute** 时需要此值。机器池的名称。	**worker**

参数	描述	值
compute.platform	使用 **compute** 时需要此值。使用此参数指定托管 worker 机器的云供应商。此参数值必须与 **controlPlane.platform** 参数值匹配。	**aws**、**azure**、**gcp**、**openstack**、**ovirt**、**vsphere** 或 **{}**
compute.replicas	要置备的计算机器数量，也称为 worker 机器。	大于或等于 **2** 的正整数。默认值为 **3**。
controlPlane	组成 control plane 的机器的配置。	**MachinePool** 对象的数组。详情请查看以下"Machine-pool"表。
controlPlane.architecture	决定池中机器的指令集合架构。目前不支持异构集群，因此所有池都必须指定相同的架构。有效值为 **amd64** （默认值）。	字符串
controlPlane.hyperthreading	是否在 control plane 机器上启用或禁用并发多线程或**超线程**。默认情况下，启用并发多线程以提高机器内核的性能。  **重要** 如果禁用并发多线程，请确保在容量规划时考虑到机器性能可能会显著降低的问题。	**Enabled** 或 **Disabled**
controlPlane.name	使用 **controlPlane** 时需要。机器池的名称。	**master**
controlPlane.platform	使用 **controlPlane** 时需要。使用此参数指定托管 control plane 机器的云供应商。此参数值必须与 **compute.platform** 参数值匹配。	**aws**、**azure**、**gcp**、**openstack**、**ovirt**、**vsphere** 或 **{}**
controlPlane.replicas	要置备的 control plane 机器数量。	唯一支持的值是 **3**，它是默认值。

参数	描述	值
credentialsMode	Cloud Credential Operator（CCO）模式。如果没有指定任何模式，CCO 会动态地尝试决定提供的凭证的功能，在支持多个模式的平台上使用 mint 模式。  **注意** 不是所有 CCO 模式都支持所有云供应商。如需有关 CCO 模式的更多信息，请参阅*Red Hat Operator 参考指南*内容中的 *Cloud Credential Operator* 条目。	**Mint**、**Passthrough**、**Manual** 或空字符串(**""**)。
fips	启用或禁用 FIPS 模式。默认为 **false**（禁用）。如果启用了 FIPS 模式，运行 OpenShift Container Platform 的 Red Hat Enterprise Linux CoreOS (RHCOS) 机器会绕过默认的 Kubernetes 加密套件，并使用由 RHCOS 提供的加密模块。  **重要** 只有在 **x86_64** 架构中的 OpenShift Container Platform 部署支持 FIPS 验证的/Modules in Process 加密库。  **注意** 如果使用 Azure File 存储，则无法启用 FIPS 模式。	**false** 或 **true**
imageContentSources	release-image 内容的源和仓库。	对象数组。包括一个 **source** 以及可选的 **mirrors**，如下表所示。
imageContentSources.source	使用 **imageContentSources** 时需要。指定用户在镜像拉取规格中引用的仓库。	字符串
imageContentSources.mirrors	指定可能还包含同一镜像的一个或多个仓库。	字符串数组

参数	描述	值
publish	如何发布或公开集群的面向用户的端点，如 Kubernetes API、OpenShift 路由。	**Internal** 或 **External**。把 **publish** 设置为 **Internal** 以部署一个私有集群，它不能被互联网访问。默认值为 **External**。

sshKey	用于验证集群机器访问的 SSH 密钥或密钥。
	 
	注意
	对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 **ssh-agent** 进程使用的 SSH 密钥。
		一个或多个密钥。例如：
		`sshKey:`
		`  <key1>`
		`  <key2>`
		`  <key3>`

4.4.5.1.4. 其他 Google Cloud Platform（GCP）配置参数

下表中描述了额外的 GCP 配置参数：

表 4.7. 额外的 GCP 参数

参数	描述	值
platform.gcp.network	要将集群部署到的现有 VPC 的名称。	字符串。
platform.gcp.region	托管集群的 GCP 区域的名称。	任何有效的区域名称，如 **us-central1**。
platform.gcp.type	GCP 机器类型。	GCP 机器类型。
platform.gcp.zones	安装程序在其中为特定 MachinePool 创建机器的可用区。	有效的 GCP 可用区 列表，如 **us-central1-a**，在一个 YAML 序列中。

参数	描述	值
platform.gcp.controlPlaneSubnet	要将 control plane 机器部署到的 VPC 中现有子网的名称。	子网名称。
platform.gcp.computeSubnet	您要将计算机器部署到的 VPC 中现有子网的名称。	子网名称。
platform.gcp.licenses	必须应用到计算镜像的许可证 URL 列表。 **重要** License **参数** 是一个已弃用的字段，嵌套虚拟化默认为启用。不建议使用此字段。	许可证 API 可获得的任何许可证，如启用 嵌套虚拟化 的许可证。您不能将此参数与生成预构建镜像的机制一起使用。使用许可证 URL 强制安装程序复制源镜像，然后再使用。
platform.gcp.osDisk.diskSizeGB	以 GB (GB) 为单位的磁盘大小。	16 GB 到 65536 GB 之间的任何大小。
platform.gcp.osDisk.diskType	磁盘类型。	默认 **pd-ssd** 或 **pd-standard** 磁盘类型。control plane 节点必须是 **pd-ssd** 磁盘类型。worker 节点可以是其中任一个类型。

4.4.5.2. GCP 的自定义 install-config.yaml 文件示例

您可以自定义 **install-config.yaml** 文件，以指定有关 OpenShift Container Platform 集群平台的更多信息，或修改所需参数的值。

重要

此示例 YAML 文件仅供参考。您必须使用安装程序来获取 **install-config.yaml** 文件，并且修改该文件。

```
apiVersion: v1
baseDomain: example.com (1)
controlPlane: (2) (3)
  hyperthreading: Enabled (4)
  name: master
```

```
  platform:
    gcp:
      type: n2-standard-4
      zones:
      - us-central1-a
      - us-central1-c
      osDisk:
        diskType: pd-ssd
        diskSizeGB: 1024
  replicas: 3
compute: 5 6
- hyperthreading: Enabled 7
  name: worker
  platform:
    gcp:
      type: n2-standard-4
      zones:
      - us-central1-a
      - us-central1-c
      osDisk:
        diskType: pd-standard
        diskSizeGB: 128
  replicas: 3
metadata:
  name: test-cluster 8
networking:
  clusterNetwork:
  - cidr: 10.128.0.0/14
    hostPrefix: 23
  machineNetwork:
  - cidr: 10.0.0.0/16
  networkType: OpenShiftSDN
  serviceNetwork:
  - 172.30.0.0/16
platform:
  gcp:
    projectID: openshift-production 9
    region: us-central1 10
pullSecret: '{"auths": ...}' 11
fips: false 12
sshKey: ssh-ed25519 AAAA... 13
```

1 8 9 10 11 必需。安装程序会提示您输入这个值。

2 5 如果没有提供这些参数和值，安装程序会提供默认值。

3 6 **controlPlane** 部分是一个单映射，但 **compute** 部分是一系列映射。为满足不同数据结构的要求，**compute** 部分的第一行必须以连字符 **-** 开头，**controlPlane** 部分的第一行则不可以连字符开头。只使用一个 control plane 池。

4 7 是否要启用或禁用并发多线程或**超线程**。默认情况下，启用并发多线程以提高机器内核的性能。您可以通过将参数值设为 **Disabled** 来禁用。如果您在某些集群机器上禁用并发多线程，则必须在所有集群机器上禁用。

重要

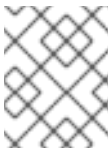
如果禁用并发多线程，请确保在容量规划时考虑到机器性能可能会显著降低的问题。如果您禁用并发多线程，请为您的机器使用较大的类型，如 **n1-standard-8**。

12 是否启用或禁用 FIPS 模式。默认情况下不启用 FIPS 模式。如果启用了 FIPS 模式，运行 OpenShift Container Platform 的 Red Hat Enterprise Linux CoreOS (RHCOS) 机器会绕过默认的 Kubernetes 加密套件，并使用由 RHCOS 提供的加密模块。

重要

只有在 **x86_64** 架构中的 OpenShift Container Platform 部署支持 FIPS 验证的/Modules in Process 加密库。

13 您可以选择提供您用来访问集群中机器的 **sshKey** 值。



注意

对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 **ssh-agent** 进程使用的 SSH 密钥。

4.4.5.3. 在安装过程中配置集群范围代理

生产环境可能会拒绝直接访问互联网，而是提供 HTTP 或 HTTPS 代理。您可以通过在 **install-config.yaml** 文件中配置代理设置，将新的 OpenShift Container Platform 集群配置为使用代理。

先决条件

- 您有一个现有的 **install-config.yaml** 文件。
- 您检查了集群需要访问的站点，并决定是否需要绕过代理。默认情况下代理所有集群出口流量，包括对托管云供应商 API 的调用。您需要将站点添加到 **Proxy** 对象的 **spec.noProxy** 字段来绕过代理。

注意

Proxy 对象 **status.noProxy** 字段使用安装配置中的 **networking.machineNetwork[].cidr**、**networking.clusterNetwork[].cidr** 和 **networking.serviceNetwork[]** 字段的值填充。

对于在 Amazon Web Services(AWS)、Google Cloud Platform(GCP)、Microsoft Azure 和 Red Hat OpenStack Platform(RHOSP)上安装, **Proxy** 对象 **status.noProxy** 字段也会使用实例元数据端点填充(**169.254.169.254**)。

流程

1. 编辑 **install-config.yaml** 文件并添加代理设置。例如：

```
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
```

```
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: example.com 3
additionalTrustBundle: | 4
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
...
```

1. 用于创建集群外 HTTP 连接的代理 URL。URL 必须是 **http**。
2. 用于创建集群外 HTTPS 连接的代理 URL。
3. 要排除在代理中的目标域名、IP 地址或其他网络 CIDR 的逗号分隔列表。在域前面加 **.** 来仅匹配子域。例如： **.y.com** 匹配 **x.y.com**，但不匹配 **y.com**。使用 * 绕过所有目的地的代理。
4. 如果提供，安装程序会在 **openshift-config 命名空间中生成名为 user-ca- bundle** 的配置映射来保存额外的 CA 证书。如果您提供 **additionalTrustBundle** 和至少一个代理设置，则 **Proxy** 对象会被配置为引用 **trustedCA** 字段中的 **user-ca-bundle** 配置映射。然后，Cluster Network Operator 会创建一个 **trusted-ca-bundle** 配置映射，该配置映射将为 **trustedCA** 参数指定的内容与 RHCOS 信任捆绑包合并。 **additionalTrustBundle** 字段是必需的，除非代理的身份证书由来自 RHCOS 信任捆绑包的颁发机构签名。

注意

安装程序不支持代理的 **readinessEndpoints** 字段。

2. 保存该文件，并在安装 OpenShift Container Platform 时引用。

安装程序会创建一个名为 **cluster** 的集群范围代理，该代理使用提供的 **install-config.yaml** 文件中的代理设置。如果没有提供代理设置，仍然会创建一个 **cluster Proxy** 对象，但它会有一个空 **spec**。

注意

只支持名为 **cluster** 的 **Proxy** 对象，且无法创建额外的代理。

4.4.6. 部署集群

您可以在兼容云平台中安装 OpenShift Container Platform。

重要

安装程序的 **create cluster** 命令只能在初始安装过程中运行一次。

先决条件

- 配置托管集群的云平台的帐户。
- 获取 OpenShift Container Platform 安装程序以及集群的 pull secret。

流程

1. 对于存储在以下位置的任何现有 GCP 凭证，删除不使用您为集群配置的 GCP 帐户的服务帐户密钥的凭证：

   - **GOOGLE_CREDENTIALS**、**GOOGLE_CLOUD_KEYFILE_JSON** 或 **GCLOUD_KEYFILE_JSON** 环境变量
   - **~/.gcp/osServiceAccount.json** 文件
   - **gcloud cli** 默认凭证

2. 更改为包含安装程序的目录并初始化集群部署：

```
$ ./openshift-install create cluster --dir <installation_directory> \ 1
    --log-level=info 2
```

1 对于 **<installation_directory>**，请指定自定义 **./install-config.yaml** 文件的位置。

2 要查看不同的安装详情，请指定 **warn**、**debug** 或 **error**，而不要指定 **info**。

注意

如果您在主机上配置的云供应商帐户没有足够的权限来部署集群，安装过程将会停止，并且显示缺少的权限。

集群部署完成后，终端会显示访问集群的信息，包括指向其 Web 控制台的链接和 **kubeadmin** 用户的凭证。

输出示例

```
...
INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run 'export
KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'
INFO Access the OpenShift web-console here: https://console-openshift-
console.apps.mycluster.example.com
INFO Login to the console with user: "kubeadmin", and password: "4vYBz-Ee6gm-ymBZj-
Wt5AL"
INFO Time elapsed: 36m22s
```

注意

当安装成功时，集群访问和凭证信息还会输出到 **<installation_directory>/.openshift_install.log**。

重要

- 安装程序生成的 Ignition 配置文件包含在 24 小时后过期的证书，然后在过期时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外情况是，您需要手动批准待处理的 **node-bootstrapper** 证书签名请求（CSR）来恢复 kubelet 证书。如需更多信息，请参阅*从过期的 control plane 证书中恢复*的文档。
- 建议您在生成 12 小时后使用 Ignition 配置文件，因为集群安装后 24 小时证书从 16 小时轮转至 22 小时。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中运行证书更新时避免安装失败。

重要

您不得删除安装程序或安装程序所创建的文件。需要这两者才能删除集群。

3. 可选：您可以减少用于安装集群的服务帐户的权限数。
   - 如果将 **Owner** 角色分配给您的服务帐户，您可以删除该角色并使用 **Viewer** 角色替换。
   - 如果您包含 **Service Account Key Admin** 角色，您可以将其删除。

4.4.7. 通过下载二进制文件安装 OpenShift CLI

您需要安装 CLI（**oc**） 来使用命令行界面与 OpenShift Container Platform 进行交互。您可在 Linux、Windows 或 macOS 上安装 **oc**。

重要

如果安装了旧版本的 **oc**，则无法使用 OpenShift Container Platform 4.6 中的所有命令。下载并安装新版本的 **oc**。

4.4.7.1. 在 Linux 上安装 OpenShift CLI

您可以按照以下流程在 Linux 上安装 OpenShift CLI（**oc**）二进制文件。

流程

1. 进入到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 在 **Version** 下拉菜单中选择相应的版本。
3. 单击 **OpenShift v4.6 Linux 客户端** 条目旁边的 **Download Now**，再保存文件。
4. 解包存档：

```
$ tar xvzf <file>
```

5. 把 **oc** 二进制代码放到 **PATH** 中的目录中。
   执行以下命令可以查看当前的 **PATH** 设置：

```
$ echo $PATH
```

安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
$ oc <command>
```

4.4.7.2. 在 Windows 上安装 OpenShift CLI

您可以按照以下流程在 Windows 上安装 OpenShift CLI（**oc**）二进制代码。

流程

1. 进入到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 在 **Version** 下拉菜单中选择相应的版本。
3. 单击 **OpenShift v4.6 Windows 客户端**条目旁边的 **Download Now**，再保存文件。
4. 使用 ZIP 程序解压存档。
5. 把 **oc** 二进制代码放到 **PATH** 中的目录中。
   要查看您的 **PATH**，请打开命令提示窗口并执行以下命令：

   ```
   C:\> path
   ```

安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
C:\> oc <command>
```

4.4.7.3. 在 macOS 上安装 OpenShift CLI

您可以按照以下流程在 macOS 上安装 OpenShift CLI（**oc**）二进制代码。

流程

1. 进入到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 在 **Version** 下拉菜单中选择相应的版本。
3. 单击 **OpenShift v4.6 MacOSX 客户端**条目旁边的 **Download Now**，再保存文件。
4. 解包和解压存档。
5. 将 **oc** 二进制文件移到 PATH 的目录中。
   要查看您的 **PATH**，打开一个终端窗口并执行以下命令：

   ```
   $ echo $PATH
   ```

安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
$ oc <command>
```

4.4.8. 使用 CLI 登录到集群

您可以通过导出集群 **kubeconfig** 文件，以默认系统用户身份登录集群。**kubeconfig** 文件包含关于集群的信息，供 CLI 用于将客户端连接到正确集群和 API 服务器。该文件特只适用于一个特定的集群，在 OpenShift Container Platform 安装过程中创建。

先决条件

- 已部署了 OpenShift Container Platform 集群。
- 已安装 **oc** CLI。

流程

1. 导出 **kubeadmin** 凭证：

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

2. 使用导出的配置，验证能否成功运行 **oc** 命令：

```
$ oc whoami
```

输出示例

```
system:admin
```

其他资源

- 如需有关访问和了解 OpenShift Container Platform Web 控制台的更多信息，请参阅访问 Web 控制台。

4.4.9. OpenShift Container Platform 的 Telemetry 访问

在 OpenShift Container Platform 4.6 中，默认运行的 Telemetry 服务提供有关集群健康状况和成功更新的指标，需要访问互联网。如果您的集群连接到互联网，Telemetry 会自动运行，而且集群会注册到 OpenShift Cluster Manager。

确认 OpenShift Cluster Manager 清单正确后，可以由 Telemetry 自动维护，也可以使用 OpenShift Cluster Manager 手动维护，使用订阅监控来跟踪帐户或多集群级别的 OpenShift Container Platform 订阅。

其他资源

- 有关 Telemetry 服务的更多信息，请参阅关于远程健康监控。

4.4.10. 后续步骤

- 自定义集群。
- 若有需要，您可以选择不使用远程健康报告。

4.5. 使用自定义设置在 GCP 上安装集群

在 OpenShift Container Platform 版本 4.6 中，您可以使用自定义的网络配置在 Google Cloud Platform（GCP）上的由安装程序置备的基础架构上安装集群。通过自定义网络配置，您的集群可以与环境中现有的 IP 地址分配共存，并与现有的 MTU 和 VXLAN 配置集成。要自定义安装，请在安装集群前修改 **install-config.yaml** 文件中的参数。

大部分网络配置参数必须在安装过程中设置，只有 **kubeProxy** 配置参数可以在运行的集群中修改。

4.5.1. 先决条件

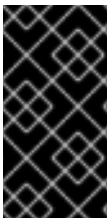
- 查看有关 OpenShift Container Platform 安装和更新流程的详细信息。
- 配置 GCP 帐户以托管集群。
- 如果使用防火墙，则必须将其配置为允许集群需要访问的站点。
- 如果不允许系统管理身份和访问管理（IAM），集群管理员可以 手动创建和维护 IAM 凭证。手动模式也可以用于云 IAM API 无法访问的环境中。

4.5.2. OpenShift Container Platform 的互联网访问

在 OpenShift Container Platform 4.6 中，您需要访问互联网来安装集群。

您必须具有以下互联网访问权限：

- 访问 OpenShift Cluster Manager 以下载安装程序并执行订阅管理。如果集群可以访问互联网，并且没有禁用 Telemetry，该服务会自动授权您的集群。
- 访问 Quay.io，以获取安装集群所需的软件包。
- 获取执行集群更新所需的软件包。



重要

如果您的集群无法直接访问互联网，则可以在置备的某些类基础架构上执行受限网络安装。在此过程中，您要下载所需的内容，并使用它在镜像 registry（mirror registry） 中填充安装集群并生成安装程序所需的软件包。对于某些安装类型，集群要安装到的环境不需要访问互联网。在更新集群之前，要更新 registry 镜像系统中的内容。

4.5.3. 生成 SSH 私钥并将其添加到代理中

如果要在集群上执行安装调试或灾难恢复，则必须为 **ssh-agent** 和安装程序提供 SSH 密钥。您可以使用此密钥访问公共集群中的 bootstrap 机器来排除安装问题。

注意

在生产环境中，您需要进行灾难恢复和调试。

您可以使用此密钥以 **core** 用户身份通过 SSH 连接到 master 节点。在部署集群时，此密钥会添加到 **core** 用户的 **~/.ssh/authorized_keys** 列表中。

注意

您必须使用一个本地密钥，而不要使用在特定平台上配置的密钥，如 AWS 密钥对。

流程

1. 如果还没有为计算机上免密码身份验证而配置的 SSH 密钥，请创建一个。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ ssh-keygen -t ed25519 -N '' \
  -f <path>/<file_name> 1
```

1 指定新 SSH 密钥的路径和文件名，如 **~/.ssh/id_rsa**。如果您已有密钥对，请确保您的公钥位于 **~/.ssh** 目录中。

运行此命令会在指定的位置生成不需要密码的 SSH 密钥。

注意

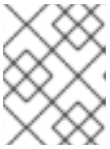
如果您计划在 **x86_64** 架构中安装使用 FIPS 验证的/Modules in Process 加密库的 OpenShift Container Platform 集群，不要创建使用 **ed25519** 算法的密钥。反之，创建一个使用 **rsa** 或 **ecdsa** 算法的密钥。

2. 作为后台任务启动 **ssh-agent** 进程：

```
$ eval "$(ssh-agent -s)"
```

输出示例

```
Agent pid 31874
```



注意

如果您的集群采用 FIPS 模式，则只使用 FIPS 兼容算法来生成 SSH 密钥。密钥必须是 RSA 或 ECDSA。

3. 将 SSH 私钥添加到 **ssh-agent**：

```
$ ssh-add <path>/<file_name> 1
```

输出示例

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

1 指定 SSH 私钥的路径和文件名，如 **~/.ssh/id_rsa**

4. 将 **GOOGLE_APPLICATION_CREDENTIALS** 环境变量设置为服务帐户私钥文件的完整路径。

```
$ export GOOGLE_APPLICATION_CREDENTIALS="<your_service_account_file>"
```

5. 验证是否应用了凭证。

```
$ gcloud auth list
```

后续步骤

- 在安装 OpenShift Container Platform 时，为安装程序提供 SSH 公钥。

4.5.4. 获取安装程序

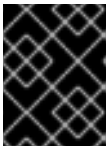
在安装 OpenShift Container Platform 之前，将安装文件下载到本地计算机上。

先决条件

- 运行 Linux 或 macOS 的计算机，本地磁盘空间为 500 MB

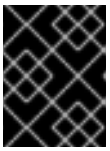
流程

1. 访问 OpenShift Cluster Manager 站点的 Infrastructure Provider 页面。如果您有红帽帐号，请使用自己的凭证登录。如果没有，请创建一个帐户。
2. 选择您的基础架构供应商。
3. 进入适用于您的安装类型的页面，下载您的操作系统的安装程序，并将文件放在要保存安装配置文件的目录中。。



重要

安装程序会在用来安装集群的计算机上创建若干文件。在完成集群安装后，您必须保留安装程序和安装程序所创建的文件。这两个文件都需要删除集群。



重要

删除安装程序创建的文件不会删除您的集群，即使集群在安装过程中失败也是如此。要删除集群，为特定云供应商完成 OpenShift Container Platform 卸载流程。

4. 提取安装程序。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ tar xvf openshift-install-linux.tar.gz
```

5. 从 Red Hat OpenShift Cluster Manager 下载安装 pull secret 。通过此 pull secret，您可以进行所含授权机构提供的服务的身份验证，这些服务包括为 OpenShift Container Platform 组件提供容器镜像的 Quay.io。

4.5.5. 创建安装配置文件

您可以自定义在 Google Cloud Platform (GCP) 上安装的 OpenShift Container Platform 集群。

先决条件

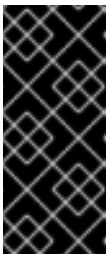
- 获取 OpenShift Container Platform 安装程序以及集群的 pull secret。

流程

1. 创建 **install-config.yaml** 文件。
   a. 更改到包含安装程序的目录，再运行以下命令：

```
$ ./openshift-install create install-config --dir <installation_directory> 1
```

1 对于 **<installation_directory>**，请指定用于保存安装程序所创建的文件的目录名称。



重要

指定一个空目录。一些安装信息，如 bootstrap X.509 证书，有较短的过期间隔，因此不要重复使用安装目录。如果要重复使用另一个集群安装中的个别文件，可以将其复制到您的目录中。但是，一些安装数据的文件名可能会在发行版本之间有所改变。从 OpenShift Container Platform 老版本中复制安装文件时要格外小心。

b. 在提示符处，提供您的云的配置详情：

i. 可选：选择用来访问集群机器的 SSH 密钥。

注意

对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 **ssh-agent** 进程使用的 SSH 密钥。

ii. 选择 **gcp** 作为目标平台。

iii. 如果您没有为计算机上的 GCP 帐户配置服务帐户密钥，则必须从 GCP 获取，并粘贴文件的内容或输入文件的绝对路径。

iv. 选择要在其中置备集群的项目 ID。默认值由您配置的服务帐户指定。

v. 选择要在其中部署集群的区域。

vi. 选择集群要部署到的基域。基域与您为集群创建的公共 DNS 区对应。

vii. 为集群输入一个描述性名称。

viii. 粘贴 Red Hat OpenShift Cluster Manager 中的 pull secret 。

2. 修改 **install-config.yaml** 文件。您可以在**安装配置参数**部分中找到有关可用参数的更多信息。

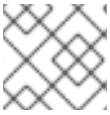
3. 备份 **install-config.yaml** 文件，以便用于安装多个集群。

重要

install-config.yaml 文件会在安装过程中消耗掉。如果要重复使用此文件，必须现在备份。

4.5.5.1. 安装配置参数

在部署 OpenShift Container Platform 集群前，您可以提供参数值，以描述托管集群的云平台的帐户并选择性地自定义集群平台。在创建 **install-config.yaml** 安装配置文件时，您可以通过命令行来提供所需的参数的值。如果要自定义集群，可以修改 **install-config.yaml** 文件来提供关于平台的更多信息。



注意

安装之后，您无法修改 **install-config.yaml** 文件中的这些参数。

重要

openshift-install 命令不验证参数的字段名称。如果指定了不正确的名称，则不会创建相关的文件或对象，且不会报告错误。确保所有指定的参数的字段名称都正确。

4.5.5.1.1. 所需的配置参数

下表描述了所需的安装配置参数：

表 4.8. 所需的参数

参数	描述	值
apiVersion	**install-config.yaml** 内容的 API 版本。当前版本是 **v1**。安装程序还可能支持旧的 API 版本。	字符串
baseDomain	云供应商的基域。此基础域用于创建到 OpenShift Container Platform 集群组件的路由。集群的完整 DNS 名称是 **baseDomain** 和 **metadata.name** 参数值的组合，其格式为 **<metadata.name>.<baseDomain>**。	完全限定域名或子域名，如 **example.com**。
metadata	Kubernetes 资源 **ObjectMeta**，其中只消耗 **name** 参数。	对象
metadata.name	集群的名称。集群的 DNS 记录是 **{{.metadata.name}}.{{.baseDomain}}** 的子域。	小写字母,连字符(-)和句点(.)的字符串，如 **dev**。
platform	执行安装的具体平台配置：**aws**、**baremetal**、**azure**、**openstack**、**ovirt**、**vsphere**。有关 **platform.<platform>** 参数的额外信息，请参考下表来了解您的具体平台。	对象

参数	描述	值

pullSecret	从 Red Hat OpenShift Cluster Manager 获取 pull secret，验证从 Quay.io 等服务中下载 OpenShift Container Platform 组件的容器镜像。	<pre>{
		  "auths":{
		  "cloud.openshift.com":{
		  "auth":"b3Blb=",
		  "email":"you@example.com"
		  },
		  "quay.io":{
		  "auth":"b3Blb=",
		  "email":"you@example.com"
		  }
		  }
		}</pre>

4.5.5.1.2. 网络配置参数

您可以根据现有网络基础架构的要求自定义安装配置。例如，您可以扩展集群网络的 IP 地址块，或者提供不同于默认值的不同 IP 地址块。

只支持 IPv4 地址。

表 4.9. 网络参数

参数	描述	值
networking	集群网络的配置。	对象 **注意** 您不能在安装后修改 **networking** 对象指定的参数。
networking.networkType	要安装的集群网络供应商 Container Network Interface（CNI）插件。	**OpenShiftSDN** 或 **OVNKubernetes**。默认值为 **OpenShiftSDN**。

networking.clusterNetwork	pod 的 IP 地址块。
	
	默认值为 **10.128.0.0/14**，主机前缀为 **/23**。
	
	如果您指定多个 IP 地址块，则块不得互相重叠。
		一个对象数组。例如：
		
		<pre>networking:
		  clusterNetwork:
		  - cidr: 10.128.0.0/14
		  hostPrefix: 23</pre>

参数	描述	值
networking.clusterNetwork.cidr	使用 **networking.clusterNetwork** 时需要此项。IP 地址块。 一个 IPv4 网络。	使用 CIDR 形式的 IP 地址块。IPv4 块的前缀长度介于 **0** 到 **32** 之间。
networking.clusterNetwork.hostPrefix	分配给每个单独节点的子网前缀长度。例如，如果 **hostPrefix** 设为 **23**，则每个节点从所给的 **cidr** 中分配一个 **/23** 子网。**hostPrefix** 值 **23** 提供 510（2^(32 - 23)- 2）个 pod IP 地址。	子网前缀。 默认值为 **23**。

networking.serviceNetwork	服务的 IP 地址块。默认值为 **172.30.0.0/16**。
	
	OpenShift SDN 和 OVN-Kubernetes 网络供应商只支持服务网络的一个 IP 地址块。
		CIDR 格式具有 IP 地址块的数组。例如：
		
		`networking:`
		`  serviceNetwork:`
		`  - 172.30.0.0/16`

networking.machineNetwork	机器的 IP 地址块。
	
	如果您指定多个 IP 地址块，则块不得互相重叠。
		一个对象数组。例如：
		
		`networking:`
		`  machineNetwork:`
		`  - cidr: 10.0.0.0/16`

networking.machineNetwork.cidr	使用 **networking.machineNetwork** 时需要。IP 地址块。libvirt 以外的所有平台的默认值为 **10.0.0.0/16**。对于 libvirt，默认值为 **192.168.126.0/24**。	CIDR 表示法中的 IP 网络块。 例如： **10.0.0.0/16**。 **注意** 将 **networking.machineNetwork** 设置为与首选 NIC 所在的 CIDR 匹配。

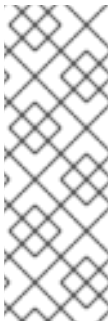
4.5.5.1.3. 可选配置参数

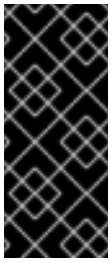
下表描述了可选安装配置参数：

表 4.10. 可选参数

参数	描述	值
additionalTrustBundle	添加到节点可信证书存储中的 PEM 编码 X.509 证书捆绑包。配置了代理时，也可以使用这个信任捆绑包。	字符串

参数	描述	值
compute	组成计算节点的机器的配置。	machine-pool 对象的数组。详情请查看以下"Machine-pool"表。
compute.architecture	决定池中机器的指令集合架构。目前不支持异构集群，因此所有池都必须指定相同的架构。有效值为 **amd64** （默认值）。	字符串
compute.hyperthreading	是否在计算机器上启用或禁用并发多线程或**超线程**。默认情况下，启用并发多线程以提高机器内核的性能。  **重要** 如果禁用并发多线程，请确保在容量规划时考虑到机器性能可能会显著降低的问题。	**Enabled** 或 **Disabled**
compute.name	使用 **compute** 时需要此值。机器池的名称。	**worker**
compute.platform	使用 **compute** 时需要此值。使用此参数指定托管 worker 机器的云供应商。此参数值必须与 **controlPlane.platform** 参数值匹配。	**aws**、**azure**、**gcp**、**openstack**、**ovirt**、**vsphere** 或 **{}**
compute.replicas	要置备的计算机器数量，也称为 worker 机器。	大于或等于 **2** 的正整数。默认值为 **3**。
controlPlane	组成 control plane 的机器的配置。	**MachinePool** 对象的数组。详情请查看以下"Machine-pool"表。
controlPlane.architecture	决定池中机器的指令集合架构。目前不支持异构集群，因此所有池都必须指定相同的架构。有效值为 **amd64** （默认值）。	字符串

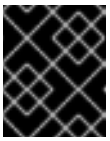
参数	描述	值
controlPlane.hyperthreading	是否在 control plane 机器上启用或禁用并发多线程或**超线程**。默认情况下，启用并发多线程以提高机器内核的性能。  **重要** 如果禁用并发多线程，请确保在容量规划时考虑到机器性能可能会显著降低的问题。	**Enabled** 或 **Disabled**
controlPlane.name	使用 **controlPlane** 时需要。机器池的名称。	**master**
controlPlane.platform	使用 **controlPlane** 时需要。使用此参数指定托管 control plane 机器的云供应商。此参数值必须与 **compute.platform** 参数值匹配。	**aws**、**azure**、**gcp**、**openstack**、**ovirt**、**vsphere** 或 **{}**
controlPlane.replicas	要置备的 control plane 机器数量。	唯一支持的值是 **3**，它是默认值。
credentialsMode	Cloud Credential Operator（CCO）模式。如果没有指定任何模式，CCO 会动态地尝试决定提供的凭证的功能，在支持多个模式的平台上使用 mint 模式。  **注意** 不是所有 CCO 模式都支持所有云供应商。如需有关 CCO 模式的更多信息，请参阅*Red Hat Operator 参考指南*内容中的 *Cloud Credential Operator* 条目。	**Mint**、**Passthrough**、**Manual** 或空字符串(**""**)。

参数	描述	值
fips	启用或禁用 FIPS 模式。默认为 **false**（禁用）。如果启用了 FIPS 模式，运行 OpenShift Container Platform 的 Red Hat Enterprise Linux CoreOS (RHCOS) 机器会绕过默认的 Kubernetes 加密套件，并使用由 RHCOS 提供的加密模块。  **重要** 只有在 **x86_64** 架构中的 OpenShift Container Platform 部署支持 FIPS 验证的/Modules in Process 加密库。  **注意** 如果使用 Azure File 存储，则无法启用 FIPS 模式。	**false** 或 **true**
imageContentSources	release-image 内容的源和仓库。	对象数组。包括一个 **source** 以及可选的 **mirrors**，如下表所示。
imageContentSources.source	使用 **imageContentSources** 时需要。指定用户在镜像拉取规格中引用的仓库。	字符串
imageContentSources.mirrors	指定可能还包含同一镜像的一个或多个仓库。	字符串数组
publish	如何发布或公开集群的面向用户的端点，如 Kubernetes API、OpenShift 路由。	**Internal** 或 **External**。把 **publish** 设置为 **Internal** 以部署一个私有集群，它不能被互联网访问。默认值为 **External**。
sshKey	用于验证集群机器访问的 SSH 密钥或密钥。  **注意** 对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 **ssh-agent** 进程使用的 SSH 密钥。	一个或多个密钥。例如： sshKey: <key1> <key2> <key3>

4.5.5.1.4. 其他 Google Cloud Platform（GCP）配置参数

下表中描述了额外的 GCP 配置参数：

表 4.11. 额外的 GCP 参数

参数	描述	值
platform.gcp.network	要将集群部署到的现有 VPC 的名称。	字符串。
platform.gcp.region	托管集群的 GCP 区域的名称。	任何有效的区域名称，如 **us-central1**。
platform.gcp.type	GCP 机器类型。	GCP 机器类型。
platform.gcp.zones	安装程序在其中为特定 MachinePool 创建机器的可用区。	有效的 GCP 可用区 列表，如 **us-central1-a**，在一个 YAML 序列中。
platform.gcp.controlPlaneSubnet	要将 control plane 机器部署到的 VPC 中现有子网的名称。	子网名称。
platform.gcp.computeSubnet	您要将计算机器部署到的 VPC 中现有子网的名称。	子网名称。
platform.gcp.licenses	必须应用到计算镜像的许可证 URL 列表。  **重要** License **参数** 是一个已弃用的字段，嵌套虚拟化默认为启用。不建议使用此字段。	许可证 API 可获得的任何许可证，如启用 嵌套虚拟化 的许可证。您不能将此参数与生成预构建镜像的机制一起使用。使用许可证 URL 强制安装程序复制源镜像，然后再使用。
platform.gcp.osDisk.diskSizeGB	以 GB (GB) 为单位的磁盘大小。	16 GB 到 65536 GB 之间的任何大小。

参数	描述	值
platform.gcp.osDisk.diskType	磁盘类型。	默认 **pd-ssd** 或 **pd-standard** 磁盘类型。control plane 节点必须是 **pd-ssd** 磁盘类型。worker 节点可以是其中任一个类型。

4.5.5.2. GCP 的自定义 install-config.yaml 文件示例

您可以自定义 **install-config.yaml** 文件，以指定有关 OpenShift Container Platform 集群平台的更多信息，或修改所需参数的值。

重要

此示例 YAML 文件仅供参考。您必须使用安装程序来获取 **install-config.yaml** 文件，并且修改该文件。

```
apiVersion: v1
baseDomain: example.com (1)
controlPlane: (2) (3)
  hyperthreading: Enabled (4)
  name: master
  platform:
    gcp:
      type: n2-standard-4
      zones:
      - us-central1-a
      - us-central1-c
      osDisk:
        diskType: pd-ssd
        diskSizeGB: 1024
  replicas: 3
compute: (5) (6)
- hyperthreading: Enabled (7)
  name: worker
  platform:
    gcp:
      type: n2-standard-4
      zones:
      - us-central1-a
      - us-central1-c
      osDisk:
        diskType: pd-standard
        diskSizeGB: 128
  replicas: 3
metadata:
  name: test-cluster (8)
networking: (9)
  clusterNetwork:
  - cidr: 10.128.0.0/14
    hostPrefix: 23
```

```
  machineNetwork:
  - cidr: 10.0.0.0/16
  networkType: OpenShiftSDN
  serviceNetwork:
  - 172.30.0.0/16
platform:
  gcp:
    projectID: openshift-production 10
    region: us-central1 11
pullSecret: '{"auths": ...}' 12
fips: false 13
sshKey: ssh-ed25519 AAAA... 14
```

1 8 10 11 12 必需。安装程序会提示您输入这个值。

2 5 9 如果没有提供这些参数和值，安装程序会提供默认值。

3 6 **controlPlane** 部分是一个单映射，但 **compute** 部分是一系列映射。为满足不同数据结构的要求，**compute** 部分的第一行必须以连字符 **-** 开头，**controlPlane** 部分的第一行则不可以连字符开头。只使用一个 control plane 池。

4 7 是否要启用或禁用并发多线程或**超线程**。默认情况下，启用并发多线程以提高机器内核的性能。您可以通过将参数值设为 **Disabled** 来禁用。如果您在某些集群机器上禁用并发多线程，则必须在所有集群机器上禁用。

重要

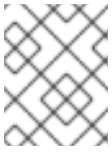
如果禁用并发多线程，请确保在容量规划时考虑到机器性能可能会显著降低的问题。如果您禁用并发多线程，请为您的机器使用较大的类型，如 **n1-standard-8**。

13 是否启用或禁用 FIPS 模式。默认情况下不启用 FIPS 模式。如果启用了 FIPS 模式，运行 OpenShift Container Platform 的 Red Hat Enterprise Linux CoreOS (RHCOS) 机器会绕过默认的 Kubernetes 加密套件，并使用由 RHCOS 提供的加密模块。

重要

只有在 **x86_64** 架构中的 OpenShift Container Platform 部署支持 FIPS 验证的/Modules in Process 加密库。

14 您可以选择提供您用来访问集群中机器的 **sshKey** 值。



注意

对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 **ssh-agent** 进程使用的 SSH 密钥。

4.5.5.3. 在安装过程中配置集群范围代理

生产环境可能会拒绝直接访问互联网，而是提供 HTTP 或 HTTPS 代理。您可以通过在 **install-config.yaml** 文件中配置代理设置，将新的 OpenShift Container Platform 集群配置为使用代理。

先决条件

- 您有一个现有的 **install-config.yaml** 文件。
- 您检查了集群需要访问的站点，并决定是否需要绕过代理。默认情况下代理所有集群出口流量，包括对托管云供应商 API 的调用。您需要将站点添加到 **Proxy** 对象的 **spec.noProxy** 字段来绕过代理。

注意

Proxy 对象 **status.noProxy** 字段使用安装配置中的 **networking.machineNetwork[].cidr**、**networking.clusterNetwork[].cidr** 和 **networking.serviceNetwork[]** 字段的值填充。

对于在 Amazon Web Services(AWS)、Google Cloud Platform(GCP)、Microsoft Azure 和 Red Hat OpenStack Platform(RHOSP)上安装，**Proxy** 对象 **status.noProxy** 字段也会使用实例元数据端点填充(**169.254.169.254**)。

流程

1. 编辑 **install-config.yaml** 文件并添加代理设置。例如：

```
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: example.com 3
additionalTrustBundle: | 4
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
...
```

1. 用于创建集群外 HTTP 连接的代理 URL。URL 必须是 **http**。
2. 用于创建集群外 HTTPS 连接的代理 URL。
3. 要排除在代理中的目标域名、IP 地址或其他网络 CIDR 的逗号分隔列表。在域前面加 **.** 来仅匹配子域。例如：**.y.com** 匹配 **x.y.com**，但不匹配 **y.com**。使用 * 绕过所有目的地的代理。
4. 如果提供，安装程序会在 **openshift-config 命名空间中生成名为 user-ca- bundle** 的配置映射来保存额外的 CA 证书。如果您提供 **additionalTrustBundle** 和至少一个代理设置，则 **Proxy** 对象会被配置为引用 **trustedCA** 字段中的 **user-ca-bundle** 配置映射。然后，Cluster Network Operator 会创建一个 **trusted-ca-bundle** 配置映射，该配置映射将为 **trustedCA** 参数指定的内容与 RHCOS 信任捆绑包合并。**additionalTrustBundle** 字段是必需的，除非代理的身份证书由来自 RHCOS 信任捆绑包的颁发机构签名。

注意

安装程序不支持代理的 **readinessEndpoints** 字段。

2. 保存该文件，并在安装 OpenShift Container Platform 时引用。

安装程序会创建一个名为 **cluster** 的集群范围代理，该代理使用提供的 **install-config.yaml** 文件中的代理设置。如果没有提供代理设置，仍然会创建一个 **cluster Proxy** 对象，但它会有一个空 **spec**。

注意

只支持名为 **cluster** 的 **Proxy** 对象，且无法创建额外的代理。

4.5.6. 网络配置阶段

当在安装前指定集群配置时，在安装过程中的几个阶段可以修改网络配置：

阶段 1

输入 **openshift-install create install-config** 命令后。在 **install-config.yaml** 文件中，您可以自定义以下与网络相关的字段：

- **networking.networkType**
- **networking.clusterNetwork**
- **networking.serviceNetwork**
- **networking.machineNetwork**
  有关这些字段的更多信息，请参阅"安装配置参数"。

注意

将 **networking.machineNetwork** 设置为与首选 NIC 所在的 CIDR 匹配。

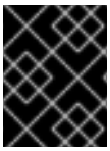
阶段 2

输入 **openshift-install create manifests** 命令后。如果必须指定高级网络配置，在这个阶段中，只能使用您要修改的字段来定义自定义的 Cluster Network Operator 清单。

在 2 阶段，您无法覆盖 **install-config.yaml** 文件中的 1 阶段中指定的值。但是，您可以在第 2 阶段进一步自定义集群网络供应商。

4.5.7. 指定高级网络配置

您可以通过为集群网络供应商指定额外的配置，使用高级配置自定义将集群整合到现有网络环境中。您只能在安装集群前指定高级网络配置。



重要

不支持修改安装程序创建的 OpenShift Container Platform 清单文件。支持应用您创建的清单文件，如以下流程所示。

先决条件

- 创建 **install-config.yaml** 文件并完成对其所做的任何修改。

流程

1. 进入包含安装程序的目录并创建清单：

```
$ ./openshift-install create manifests --dir <installation_directory>
```

其中：

<installation_directory>
指定包含集群的 **install-config.yaml** 文件的目录名称。

2. 在 **<installation_directory>/manifests/** 目录下，为高级网络配置创建一个名为 **cluster-network-03-config.yml** 的 stub 清单文件：

```
$ cat <<EOF > <installation_directory>/manifests/cluster-network-03-config.yml
apiVersion: operator.openshift.io/v1
kind: Network
metadata:
  name: cluster
spec:
EOF
```

其中：

<installation_directory>
指定包含集群的 **manifests/** 目录的目录名称。

3. 在编辑器中打开 **cluster-network-03-config.yml** 文件，并为集群指定高级网络配置，如下例所示：

为 OpenShift SDN 网络供应商指定不同的 VXLAN 端口

```
apiVersion: operator.openshift.io/v1
kind: Network
metadata:
  name: cluster
spec:
  defaultNetwork:
    openshiftSDNConfig:
      vxlanPort: 4800
```

4. 保存 **cluster-network-03-config.yml** 文件，再退出文本编辑器。

5. 可选：备份 **manifests/cluster-network-03-config.yml** 文件。创建集群时，安装程序会删除 **manifests/** 目录。

4.5.8. Cluster Network Operator 配置

集群网络的配置作为 Cluster Network Operator (CNO) 配置的一部分被指定，并存储在名为 **cluster**的自定义资源（CR）对象中。CR 指定 **operator.openshift.io** API 组中的 **Network** API 的字段。

CNO 配置会在集群安装过程中从 **Network.config.openshift.io** API 组中的 **Network** API 继承以下字段，这些字段无法更改：

clusterNetwork
从中分配 pod IP 地址的 IP 地址池。

serviceNetwork

服务的 IP 地址池。

defaultNetwork.type

集群网络供应商，如 OpenShift SDN 或 OVN-Kubernetes。

您可以通过在名为 **cluster** 的 CNO 对象中设置 **defaultNetwork** 对象的字段来为集群指定集群网络供应商配置。

4.5.8.1. Cluster Network Operator 配置对象

Cluster Network Operator（CNO）的字段在下表中描述：

表 4.12. Cluster Network Operator 配置对象

字段	类型	描述
metadata.name	**字符串**	CNO 对象的名称。这个名称始终是 **cluster**。

spec.clusterNetwork	**数组**	用于指定从哪些 IP 地址块分配 Pod IP 地址以及分配给集群中每个节点的子网前缀长度的列表。例如：
		`spec:`
		`  clusterNetwork:`
		`  - cidr: 10.128.0.0/19`
		`  hostPrefix: 23`
		`  - cidr: 10.128.32.0/19`
		`  hostPrefix: 23`
		此值是只读的，并在 **install-config.yaml** 文件中指定。

spec.serviceNetwork	**数组**	服务的 IP 地址块。OpenShift SDN 和 OVN-Kubernetes Container Network Interface（CNI）网络供应商只支持服务网络具有单个 IP 地址块。例如：
		`spec:`
		`  serviceNetwork:`
		`  - 172.30.0.0/14`
		此值是只读的，并在 **install-config.yaml** 文件中指定。

spec.defaultNetwork	**对象**	为集群网络配置 Container Network Interface（CNI）集群网络供应商。
spec.kubeProxyConfig	**对象**	此对象的字段指定 kube-proxy 配置。如果您使用 OVN-Kubernetes 集群网络供应商，则 kube-proxy 的配置不会起作用。

defaultNetwork 对象配置
defaultNetwork 对象的值在下表中定义：

表 4.13. defaultNetwork 对象

字段	类型	描述
type	**字符串**	**OpenShiftSDN** 或 **OVNKubernetes**。在安装过程中选择了集群网络供应商。集群安装后无法更改这个值。  **注意** OpenShift Container Platform 默认使用 OpenShift SDN Container Network Interface（CNI）集群网络供应商。
openshiftSDNConfig	**对象**	此对象仅对 OpenShift SDN 集群网络供应商有效。
ovnKubernetesConfig	**对象**	此对象仅对 OVN-Kubernetes 集群网络供应商有效。

配置 OpenShift SDN CNI 集群网络供应商

下表描述了 OpenShift SDN Container Network Interface（CNI）集群网络供应商的配置字段。

表 4.14. openshiftSDNConfig 对象

字段	类型	描述
mode	**字符串**	配置 OpenShift SDN 的网络隔离模式。默认值为 **NetworkPolicy**。 **Multitenant** 和 **Subnet** 的值可以向后兼容 OpenShift Container Platform 3.x，但不推荐这样做。集群安装后无法更改这个值。
mtu	**整数**	VXLAN 覆盖网络的最大传输单元 (MTU) 。这根据主网络接口的 MTU 自动探测。您通常不需要覆盖检测到的 MTU。 如果自动探测的值不是您期望的，请确认节点上主网络接口中的 MTU 是正确的。您不能使用这个选项更改节点上主网络接口的 MTU 值。 如果您的集群中的不同节点需要不同的 MTU 值，则必须将此值设置为比集群中的最低 MTU 值小 **50**。例如，如果集群中的某些节点的 MTU 为 **9001**，而某些节点的 MTU 为 **1500**，则必须将此值设置为 **1450**。 集群安装后无法更改这个值。

字段	类型	描述
vxlanPort	**整数**	用于所有 VXLAN 数据包的端口。默认值为 **4789**。集群安装后无法更改这个值。 如果您在虚拟环境中运行，并且现有节点是另一个 VXLAN 网络的一部分，那么可能需要更改此值。例如，当在 VMware NSX-T 上运行 OpenShift SDN 覆盖时，您必须为 VXLAN 选择一个备用端口，因为两个 SDN 都使用相同的默认 VXLAN 端口号。 在 Amazon Web Services (AWS) 上，您可以在端口 **9000** 和端口 **9999** 之间为 VXLAN 选择一个备用端口。

OpenShift SDN 配置示例

```
defaultNetwork:
  type: OpenShiftSDN
  openshiftSDNConfig:
    mode: NetworkPolicy
    mtu: 1450
    vxlanPort: 4789
```

配置 OVN-Kubernetes CNI 集群网络供应商
下表描述了 OVN-Kubernetes CNI 集群网络供应商的配置字段。

表 4.15. ovnKubernetesConfig 对象

字段	类型	描述
mtu	**整数**	Geneve（Generic Network Virtualization Encapsulation）覆盖网络的最大传输单元（MTU）。这根据主网络接口的 MTU 自动探测。您通常不需要覆盖检测到的 MTU。 如果自动探测的值不是您期望的，请确认节点上主网络接口中的 MTU 是正确的。您不能使用这个选项更改节点上主网络接口的 MTU 值。 如果您的集群中的不同节点需要不同的 MTU 值，则必须将此值设置为比集群中的最低 MTU 值小 **100**。例如，如果集群中的某些节点的 MTU 为 **9001**，而某些节点的 MTU 为 **1500**，则必须将此值设置为 **1400**。 集群安装后无法更改这个值。
genevePort	**整数**	用于所有 Geneve 数据包的端口。默认值为 **6081**。集群安装后无法更改这个值。

OVN-Kubernetes 配置示例

```
defaultNetwork:
  type: OVNKubernetes
  ovnKubernetesConfig:
```

```
    mtu: 1400
    genevePort: 6081
```

kubeProxyConfig 对象配置
kubeProxyConfig 对象的值在下表中定义：

表 4.16. kubeProxyConfig 对象

字段	类型	描述
iptablesSyncPeriod	**字符串**	**iptables** 规则的刷新周期。默认值为 **30s**。有效的后缀包括 **s**、**m** 和 **h**，具体参见 Go **time** 软件包文档。 **注意** 由于 OpenShift Container Platform 4.3 及更高版本中引进了性能上的改进，现在不再需要调整 **iptablesSyncPeriod** 参数。

proxyArguments.iptables-min-sync-period	**数组**	刷新 **iptables** 规则前的最短时长。此字段确保刷新的频率不会过于频繁。有效的后缀包括 **s**、**m** 和 **h**，具体参见 Go **time** 软件包。默认值为：
		
		`kubeProxyConfig:`
		`  proxyArguments:`
		`  iptables-min-sync-period:`
		`  - 0s`

4.5.9. 部署集群

您可以在兼容云平台中安装 OpenShift Container Platform。

重要

安装程序的 **create cluster** 命令只能在初始安装过程中运行一次。

先决条件

- 配置托管集群的云平台的帐户。
- 获取 OpenShift Container Platform 安装程序以及集群的 pull secret。

流程

1. 更改为包含安装程序的目录并初始化集群部署：

```
$ ./openshift-install create cluster --dir <installation_directory> \ 1
    --log-level=info 2
```

1 对于 **<installation_directory>**，请指定

2 要查看不同的安装详情，请指定 **warn**、**debug** 或 **error**，而不要指定 **info**。

注意

如果您在主机上配置的云供应商帐户没有足够的权限来部署集群，安装过程将会停止，并且显示缺少的权限。

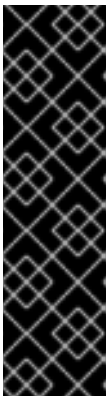
集群部署完成后，终端会显示访问集群的信息，包括指向其 Web 控制台的链接和 **kubeadmin** 用户的凭证。

输出示例

```
...
INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run 'export
KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'
INFO Access the OpenShift web-console here: https://console-openshift-
console.apps.mycluster.example.com
INFO Login to the console with user: "kubeadmin", and password: "4vYBz-Ee6gm-ymBZj-
Wt5AL"
INFO Time elapsed: 36m22s
```

注意

当安装成功时，集群访问和凭证信息还会输出到 **<installation_directory>/.openshift_install.log**。



重要

- 安装程序生成的 Ignition 配置文件包含在 24 小时后过期的证书，然后在过期时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外情况是，您需要手动批准待处理的 **node-bootstrapper** 证书签名请求（CSR）来恢复 kubelet 证书。如需更多信息，请参阅*从过期的 control plane 证书中恢复*的文档。
- 建议您在生成 12 小时后使用 Ignition 配置文件，因为集群安装后 24 小时证书从 16 小时轮转至 22 小时。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中运行证书更新时避免安装失败。

重要

您不得删除安装程序或安装程序所创建的文件。需要这两者才能删除集群。

4.5.10. 通过下载二进制文件安装 OpenShift CLI

您需要安装 CLI（**oc**）来使用命令行界面与 OpenShift Container Platform 进行交互。您可在 Linux、Windows 或 macOS 上安装 **oc**。

重要

如果安装了旧版本的 **oc**，则无法使用 OpenShift Container Platform 4.6 中的所有命令。下载并安装新版本的 **oc**。

4.5.10.1. 在 Linux 上安装 OpenShift CLI

您可以按照以下流程在 Linux 上安装 OpenShift CLI（**oc**）二进制文件。

流程

1. 进入到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 在 **Version** 下拉菜单中选择相应的版本。
3. 单击 **OpenShift v4.6 Linux 客户端** 条目旁边的 **Download Now**，再保存文件。
4. 解包存档：

```
$ tar xvzf <file>
```

5. 把 **oc** 二进制代码放到 **PATH** 中的目录中。
   执行以下命令可以查看当前的 **PATH** 设置：

```
$ echo $PATH
```

安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
$ oc <command>
```

4.5.10.2. 在 Windows 上安装 OpenShift CLI

您可以按照以下流程在 Windows 上安装 OpenShift CLI（**oc**）二进制代码。

流程

1. 进入到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 在 **Version** 下拉菜单中选择相应的版本。
3. 单击 **OpenShift v4.6 Windows 客户端**条目旁边的 **Download Now**，再保存文件。
4. 使用 ZIP 程序解压存档。
5. 把 **oc** 二进制代码放到 **PATH** 中的目录中。
   要查看您的 **PATH**，请打开命令提示窗口并执行以下命令：

```
C:\> path
```

安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
C:\> oc <command>
```

4.5.10.3. 在 macOS 上安装 OpenShift CLI

您可以按照以下流程在 macOS 上安装 OpenShift CLI（**oc**）二进制代码。

流程

1. 进入到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 在 **Version** 下拉菜单中选择相应的版本。
3. 单击 **OpenShift v4.6 MacOSX 客户端**条目旁边的 **Download Now**，再保存文件。
4. 解包和解压存档。
5. 将 **oc** 二进制文件移到 PATH 的目录中。
   要查看您的 **PATH**，打开一个终端窗口并执行以下命令：

```
$ echo $PATH
```

安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
$ oc <command>
```

4.5.11. 使用 CLI 登录到集群

您可以通过导出集群 **kubeconfig** 文件，以默认系统用户身份登录集群。**kubeconfig** 文件包含关于集群的信息，供 CLI 用于将客户端连接到正确集群和 API 服务器。该文件特只适用于一个特定的集群，在 OpenShift Container Platform 安装过程中创建。

先决条件

- 已部署了 OpenShift Container Platform 集群。
- 已安装 **oc** CLI。

流程

1. 导出 **kubeadmin** 凭证：

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

2. 使用导出的配置，验证能否成功运行 **oc** 命令：

```
$ oc whoami
```

输出示例

```
system:admin
```

其他资源

- 如需有关访问和了解 OpenShift Container Platform Web 控制台的更多信息，请参阅访问 Web 控制台。

4.5.12. OpenShift Container Platform 的 Telemetry 访问

在 OpenShift Container Platform 4.6 中，默认运行的 Telemetry 服务提供有关集群健康状况和成功更新的指标，需要访问互联网。如果您的集群连接到互联网，Telemetry 会自动运行，而且集群会注册到 OpenShift Cluster Manager。

确认 OpenShift Cluster Manager 清单正确后，可以由 Telemetry 自动维护，也可以使用 OpenShift Cluster Manager 手动维护，使用订阅监控来跟踪帐户或多集群级别的 OpenShift Container Platform 订阅。

其他资源

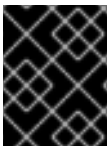
- 有关 Telemetry 服务的更多信息，请参阅关于远程健康监控。

4.5.13. 后续步骤

- 自定义集群。
- 若有需要，您可以选择不使用远程健康报告。

4.6. 在受限网络中的 GCP 上安装集群

在 OpenShift Container Platform 4.6 中，您可以通过在现有 Google Virtual Private Cloud（VPC）上创建安装发行内容的内部镜像在受限网络中的 Google Cloud Platform（GCP）上安装集群。



重要

您可以使用镜像安装发行内容安装 OpenShift Container Platform 集群，但您的集群需要访问互联网才能使用 GCP API。

4.6.1. 先决条件

- 您已为断开连接的安装 mirror 了 registry 的镜像 并获取您的 OpenShift Container Platform 版本的 **imageContentSources** 数据。

  重要

  由于安装介质位于堡垒主机上，因此请使用该计算机完成所有安装步骤。

- 在 GCP 中有一个现有的 VPC。在使用安装程序置备的基础架构的受限网络中安装集群时，您无法使用安装程序置备的 VPC。您必须使用用户置备的 VPC 来满足以下要求之一：
  - 包含镜像 registry
  - 具有防火墙规则或对等连接来访问其他位置托管的镜像 registry
- 您可以参阅有关 OpenShift Container Platform 安装和更新流程的详细信息。
- 如果使用防火墙，则必须将其配置为允许集群需要访问的站点。虽然您可能需要授予更多站点的访问权限，但您必须授予对 ***.googleapis.com** 和 **accounts.google.com** 的访问权限。
- 如果不允许系统管理身份和访问管理（IAM），集群管理员可以 手动创建和维护 IAM 凭证。手动模式也可以用于云 IAM API 无法访问的环境中。

4.6.2. 关于在受限网络中安装

在 OpenShift Container Platform 4.6 中，可以执行不需要有效的互联网连接来获取软件组件的安装。受限网络安装可使用安装程序置备的基础架构或用户置备的基础架构完成，具体取决于您要安装集群的云平台。

如果选择在云平台中执行受限网络安装，仍然需要访问其云 API。有些云功能，比如 Amazon Web Service 的 Route 53 DNS 和 IAM 服务，需要访问互联网。根据您的网络，在裸机硬件或 VMware vSphere 上安装时可能需要较少的互联网访问。

要完成受限网络安装，您必须创建一个 registry，镜像 OpenShift Container Platform registry 的内容并包含其安装介质。您可以在堡垒主机上创建此镜像，该主机可同时访问互联网和您的封闭网络，也可以使用满足您的限制条件的其他方法。

4.6.2.1. 其他限制

受限网络中的集群还有以下额外限制：

- **ClusterVersion** 状态包含一个 **Unable to retrieve available updates** 错误。
- 默认情况下，您无法使用 Developer Catalog 的内容，因为您无法访问所需的镜像流标签。

4.6.3. OpenShift Container Platform 的互联网访问

在 OpenShift Container Platform 4.6 中，您需要访问互联网来获得用来安装集群的镜像。

您必须具有以下互联网访问权限：

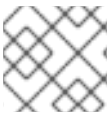
- 访问 OpenShift Cluster Manager 以下载安装程序并执行订阅管理。如果集群可以访问互联网，并且没有禁用 Telemetry，该服务会自动授权您的集群。
- 访问 Quay.io，以获取安装集群所需的软件包。
- 获取执行集群更新所需的软件包。

重要

如果您的集群无法直接访问互联网，则可以在置备的某些类基础架构上执行受限网络安装。在此过程中，您要下载所需的内容，并使用它在镜像 registry（mirror registry） 中填充安装集群并生成安装程序所需的软件包。对于某些安装类型，集群要安装到的环境不需要访问互联网。在更新集群之前，要更新 registry 镜像系统中的内容。

4.6.4. 生成 SSH 私钥并将其添加到代理中

如果要在集群上执行安装调试或灾难恢复，则必须为 **ssh-agent** 和安装程序提供 SSH 密钥。您可以使用此密钥访问公共集群中的 bootstrap 机器来排除安装问题。



注意

在生产环境中，您需要进行灾难恢复和调试。

您可以使用此密钥以 **core** 用户身份通过 SSH 连接到 master 节点。在部署集群时，此密钥会添加到 **core** 用户的 **~/.ssh/authorized_keys** 列表中。

注意

您必须使用一个本地密钥，而不要使用在特定平台上配置的密钥，如 AWS 密钥对。

流程

1. 如果还没有为计算机上免密码身份验证而配置的 SSH 密钥，请创建一个。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ ssh-keygen -t ed25519 -N '' \
  -f <path>/<file_name> 1
```

1 指定新 SSH 密钥的路径和文件名，如 **~/.ssh/id_rsa**。如果您已有密钥对，请确保您的公钥位于 **~/.ssh** 目录中。

运行此命令会在指定的位置生成不需要密码的 SSH 密钥。

注意

如果您计划在 **x86_64** 架构中安装使用 FIPS 验证的/Modules in Process 加密库的 OpenShift Container Platform 集群，不要创建使用 **ed25519** 算法的密钥。反之，创建一个使用 **rsa** 或 **ecdsa** 算法的密钥。

2. 作为后台任务启动 **ssh-agent** 进程：

```
$ eval "$(ssh-agent -s)"
```

输出示例

```
Agent pid 31874
```

注意

如果您的集群采用 FIPS 模式，则只使用 FIPS 兼容算法来生成 SSH 密钥。密钥必须是 RSA 或 ECDSA。

3. 将 SSH 私钥添加到 **ssh-agent**：

```
$ ssh-add <path>/<file_name> 1
```

输出示例

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

1 指定 SSH 私钥的路径和文件名，如 **~/.ssh/id_rsa**

4. 将 **GOOGLE_APPLICATION_CREDENTIALS** 环境变量设置为服务帐户私钥文件的完整路径。

```
$ export GOOGLE_APPLICATION_CREDENTIALS="<your_service_account_file>"
```

5. 验证是否应用了凭证。

```
$ gcloud auth list
```

后续步骤

- 在安装 OpenShift Container Platform 时，为安装程序提供 SSH 公钥。

4.6.5. 创建安装配置文件

您可以自定义在 Google Cloud Platform (GCP) 上安装的 OpenShift Container Platform 集群。

先决条件

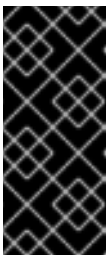
- 获取 OpenShift Container Platform 安装程序以及集群的 pull secret。对于受限网络安装，这些文件位于您的堡垒主机上。
- 具有创建镜像容器镜像仓库（registry）时生成的 **imageContentSources** 值。
- 获取您的镜像 registry 的证书内容。

流程

1. 创建 **install-config.yaml** 文件。
   a. 更改到包含安装程序的目录，再运行以下命令：

```
$ ./openshift-install create install-config --dir <installation_directory> 1
```

   1 对于 **<installation_directory>**，请指定用于保存安装程序所创建的文件的目录名称。



重要

指定一个空目录。一些安装信息，如 bootstrap X.509 证书，有较短的过期间隔，因此不要重复使用安装目录。如果要重复使用另一个集群安装中的个别文件，可以将其复制到您的目录中。但是，一些安装数据的文件名可能会在发行版本之间有所改变。从 OpenShift Container Platform 老版本中复制安装文件时要格外小心。

   b. 在提示符处，提供您的云的配置详情：
      i. 可选：选择用来访问集群机器的 SSH 密钥。

注意

对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 **ssh-agent** 进程使用的 SSH 密钥。

      ii. 选择 **gcp** 作为目标平台。
      iii. 如果您没有为计算机上的 GCP 帐户配置服务帐户密钥，则必须从 GCP 获取，并粘贴文件的内容或输入文件的绝对路径。

iv. 选择要在其中置备集群的项目 ID。默认值由您配置的服务帐户指定。

v. 选择要在其中部署集群的区域。

vi. 选择集群要部署到的基域。基域与您为集群创建的公共 DNS 区对应。

vii. 为集群输入一个描述性名称。

viii. 粘贴 Red Hat OpenShift Cluster Manager 中的 pull secret 。

2. 编辑 **install-config.yaml** 文件，以提供在受限网络中安装所需的其他信息。

a. 更新 **pullSecret** 值，使其包含 registry 的身份验证信息：

```
pullSecret: '{"auths":{"<mirror_host_name>:5000": {"auth": "<credentials>","email":
"you@example.com"}}}'
```

对于 **<mirror_host_name>**，请指定您在镜像 registry 证书中指定的 registry 域名；对于 **<credentials>**，请指定您的镜像 registry 的 base64 编码用户名和密码。

b. 添加 **additionalTrustBundle** 参数和值。

```
additionalTrustBundle: |
  -----BEGIN CERTIFICATE-----

  ZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZ
  -----END CERTIFICATE-----
```

该值必须是您用于镜像 registry 的证书文件内容，可以是现有的可信证书颁发机构或您为镜像 registry 生成的自签名证书。

c. 定义在父 **platform.gcp** 字段中安装集群的 VPC 的网络和子网：

```
network: <existing_vpc>
controlPlaneSubnet: <control_plane_subnet>
computeSubnet: <compute_subnet>
```

对于 **platform.gcp.network**，指定现有 Google VPC 的名称。对于 **platform.gcp.controlPlaneSubnet** 和 **platform.gcp.computeSubnet**，请分别指定要分别部署 control plane 机器和计算机器的现有子网。

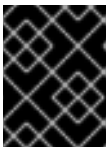
d. 添加镜像内容资源，如下例所示：

```
imageContentSources:
- mirrors:
  - <mirror_host_name>:5000/<repo_name>/release
  source: quay.example.com/openshift-release-dev/ocp-release
- mirrors:
  - <mirror_host_name>:5000/<repo_name>/release
  source: registry.example.com/ocp/release
```

要完成这些值，请使用您在创建镜像容器镜像仓库（registry）时记录的 **imageContentSources**。

3. 对需要的 **install-config.yaml** 文件做任何其他修改。您可以在**安装配置参数**部分中找到有关可用参数的更多信息。

4. 备份 **install-config.yaml** 文件，以便用于安装多个集群。



重要

install-config.yaml 文件会在安装过程中消耗掉。如果要重复使用此文件，必须现在备份。

4.6.5.1. 安装配置参数

在部署 OpenShift Container Platform 集群前，您可以提供参数值，以描述托管集群的云平台的帐户并选择性地自定义集群平台。在创建 **install-config.yaml** 安装配置文件时，您可以通过命令行来提供所需的参数的值。如果要自定义集群，可以修改 **install-config.yaml** 文件来提供关于平台的更多信息。

注意

安装之后，您无法修改 **install-config.yaml** 文件中的这些参数。

重要

openshift-install 命令不验证参数的字段名称。如果指定了不正确的名称，则不会创建相关的文件或对象，且不会报告错误。确保所有指定的参数的字段名称都正确。

4.6.5.1.1. 所需的配置参数

下表描述了所需的安装配置参数：

表 4.17. 所需的参数

参数	描述	值
apiVersion	**install-config.yaml** 内容的 API 版本。当前版本是 **v1**。安装程序还可能支持旧的 API 版本。	字符串
baseDomain	云供应商的基域。此基础域用于创建到 OpenShift Container Platform 集群组件的路由。集群的完整 DNS 名称是 **baseDomain** 和 **metadata.name** 参数值的组合，其格式为 **<metadata.name>.<baseDomain>**。	完全限定域名或子域名，如 **example.com**。
metadata	Kubernetes 资源 **ObjectMeta**，其中只消耗 **name** 参数。	对象
metadata.name	集群的名称。集群的 DNS 记录是 **{{.metadata.name}}.{{.baseDomain}}** 的子域。	小写字母,连字符(-)和句点(.)的字符串，如 **dev**。

参数	描述	值
platform	执行安装的具体平台配置：**aws**、**baremetal**、**azure**、**openstack**、**ovirt**、**vsphere**。有关 **platform.<platform>** 参数的额外信息，请参考下表来了解您的具体平台。	对象

pullSecret	从 Red Hat OpenShift Cluster Manager 获取 pull secret，验证从 Quay.io 等服务中下载 OpenShift Container Platform 组件的容器镜像。	<pre>{
		  "auths":{
		    "cloud.openshift.com":{
		      "auth":"b3Blb=",
		      "email":"you@example.com"
		    },
		    "quay.io":{
		      "auth":"b3Blb=",
		      "email":"you@example.com"
		    }
		  }
		}</pre>

4.6.5.1.2. 网络配置参数

您可以根据现有网络基础架构的要求自定义安装配置。例如，您可以扩展集群网络的 IP 地址块，或者提供不同于默认值的不同 IP 地址块。

只支持 IPv4 地址。

表 4.18. 网络参数

参数	描述	值
networking	集群网络的配置。	对象 **注意** 您不能在安装后修改 **networking** 对象指定的参数。
networking.networkType	要安装的集群网络供应商 Container Network Interface（CNI）插件。	**OpenShiftSDN** 或 **OVNKubernetes**。默认值为 **OpenShiftSDN**。

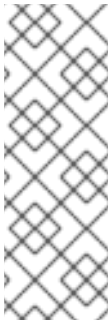
参数	描述	值
networking.clusterNetwork	pod 的 IP 地址块。 默认值为 **10.128.0.0/14**，主机前缀为 **/23**。 如果您指定多个 IP 地址块，则块不得互相重叠。	一个对象数组。例如： networking: clusterNetwork: - cidr: 10.128.0.0/14 hostPrefix: 23
networking.clusterNetwork.cidr	使用 **networking.clusterNetwork** 时需要此项。IP 地址块。 一个 IPv4 网络。	使用 CIDR 形式的 IP 地址块。IPv4 块的前缀长度介于 **0** 到 **32** 之间。
networking.clusterNetwork.hostPrefix	分配给每个单独节点的子网前缀长度。例如，如果 **hostPrefix** 设为 **23**，则每个节点从所给的 **cidr** 中分配一个 **/23** 子网。**hostPrefix** 值 **23** 提供 510（2^(32 - 23)- 2）个 pod IP 地址。	子网前缀。 默认值为 **23**。
networking.serviceNetwork	服务的 IP 地址块。默认值为 **172.30.0.0/16**。 OpenShift SDN 和 OVN-Kubernetes 网络供应商只支持服务网络的一个 IP 地址块。	CIDR 格式具有 IP 地址块的数组。例如： networking: serviceNetwork: - 172.30.0.0/16
networking.machineNetwork	机器的 IP 地址块。 如果您指定多个 IP 地址块，则块不得互相重叠。	一个对象数组。例如： networking: machineNetwork: - cidr: 10.0.0.0/16
networking.machineNetwork.cidr	使用 **networking.machineNetwork** 时需要。IP 地址块。libvirt 以外的所有平台的默认值为 **10.0.0.0/16**。对于 libvirt，默认值为 **192.168.126.0/24**。	CIDR 表示法中的 IP 网络块。 例如：**10.0.0.0/16**。 **注意** 将 **networking.machineNetwork** 设置为与首选 NIC 所在的 CIDR 匹配。

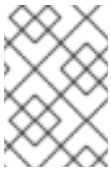
4.6.5.1.3. 可选配置参数

下表描述了可选安装配置参数：

表 4.19. 可选参数

参数	描述	值
additionalTrustBundle	添加到节点可信证书存储中的 PEM 编码 X.509 证书捆绑包。配置了代理时，也可以使用这个信任捆绑包。	字符串
compute	组成计算节点的机器的配置。	machine-pool 对象的数组。详情请查看以下"Machine-pool"表。
compute.architecture	决定池中机器的指令集合架构。目前不支持异构集群，因此所有池都必须指定相同的架构。有效值为 **amd64** （默认值）。	字符串
compute.hyperthreading	是否在计算机器上启用或禁用并发多线程或**超线程**。默认情况下，启用并发多线程以提高机器内核的性能。  **重要** 如果禁用并发多线程，请确保在容量规划时考虑到机器性能可能会显著降低的问题。	**Enabled** 或 **Disabled**
compute.name	使用 **compute** 时需要此值。机器池的名称。	**worker**
compute.platform	使用 **compute** 时需要此值。使用此参数指定托管 worker 机器的云供应商。此参数值必须与 **controlPlane.platform** 参数值匹配。	**aws**、**azure**、**gcp**、**openstack**、**ovirt**、**vsphere** 或 **{}**
compute.replicas	要置备的计算机器数量，也称为 worker 机器。	大于或等于 **2** 的正整数。默认值为 **3**。
controlPlane	组成 control plane 的机器的配置。	**MachinePool** 对象的数组。详情请查看以下"Machine-pool"表。
controlPlane.architecture	决定池中机器的指令集合架构。目前不支持异构集群，因此所有池都必须指定相同的架构。有效值为 **amd64** （默认值）。	字符串

参数	描述	值
controlPlane.hyperthreading	是否在 control plane 机器上启用或禁用并发多线程或**超线程**。默认情况下，启用并发多线程以提高机器内核的性能。  **重要** 如果禁用并发多线程，请确保在容量规划时考虑到机器性能可能会显著降低的问题。	**Enabled** 或 **Disabled**
controlPlane.name	使用 **controlPlane** 时需要。机器池的名称。	**master**
controlPlane.platform	使用 **controlPlane** 时需要。使用此参数指定托管 control plane 机器的云供应商。此参数值必须与 **compute.platform** 参数值匹配。	**aws**、**azure**、**gcp**、**openstack**、**ovirt**、**vsphere** 或 **{}**
controlPlane.replicas	要置备的 control plane 机器数量。	唯一支持的值是 **3**，它是默认值。
credentialsMode	Cloud Credential Operator（CCO）模式。如果没有指定任何模式，CCO 会动态地尝试决定提供的凭证的功能，在支持多个模式的平台上使用 mint 模式。  **注意** 不是所有 CCO 模式都支持所有云供应商。如需有关 CCO 模式的更多信息，请参阅*Red Hat Operator 参考指南*内容中的 *Cloud Credential Operator* 条目。	**Mint**、**Passthrough**、**Manual** 或空字符串(**""**)。

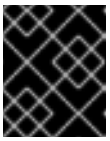
参数	描述	值
fips	启用或禁用 FIPS 模式。默认为 **false**（禁用）。如果启用了 FIPS 模式，运行 OpenShift Container Platform 的 Red Hat Enterprise Linux CoreOS (RHCOS) 机器会绕过默认的 Kubernetes 加密套件，并使用由 RHCOS 提供的加密模块。  **重要** 只有在 **x86_64** 架构中的 OpenShift Container Platform 部署支持 FIPS 验证的/Modules in Process 加密库。  **注意** 如果使用 Azure File 存储，则无法启用 FIPS 模式。	**false** 或 **true**
imageContentSources	release-image 内容的源和仓库。	对象数组。包括一个 **source** 以及可选的 **mirrors**，如下表所示。
imageContentSources.source	使用 **imageContentSources** 时需要。指定用户在镜像拉取规格中引用的仓库。	字符串
imageContentSources.mirrors	指定可能还包含同一镜像的一个或多个仓库。	字符串数组
publish	如何发布或公开集群的面向用户的端点，如 Kubernetes API、OpenShift 路由。	**Internal** 或 **External**。把 **publish** 设置为 **Internal** 以部署一个私有集群，它不能被互联网访问。默认值为 **External**。

sshKey	用于验证集群机器访问的 SSH 密钥或密钥。
	
	 
	注意
	
	对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 **ssh-agent** 进程使用的 SSH 密钥。
		一个或多个密钥。例如：
		
		`sshKey:`
		`  <key1>`
		`  <key2>`
		`  <key3>`

4.6.5.1.4. 其他 Google Cloud Platform（GCP）配置参数

下表中描述了额外的 GCP 配置参数：

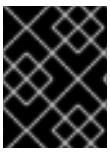
表 4.20. 额外的 GCP 参数

参数	描述	值
platform.gcp.network	要将集群部署到的现有 VPC 的名称。	字符串。
platform.gcp.region	托管集群的 GCP 区域的名称。	任何有效的区域名称，如 **us-central1**。
platform.gcp.type	GCP 机器类型。	GCP 机器类型。
platform.gcp.zones	安装程序在其中为特定 MachinePool 创建机器的可用区。	有效的 GCP 可用区 列表，如 **us-central1-a**，在一个 YAML 序列中。
platform.gcp.controlPlaneSubnet	要将 control plane 机器部署到的 VPC 中现有子网的名称。	子网名称。
platform.gcp.computeSubnet	您要将计算机器部署到的 VPC 中现有子网的名称。	子网名称。
platform.gcp.licenses	必须应用到计算镜像的许可证 URL 列表。  **重要** License **参数** 是一个已弃用的字段，嵌套虚拟化默认为启用。不建议使用此字段。	许可证 API 可获得的任何许可证，如启用 嵌套虚拟化 的许可证。您不能将此参数与生成预构建镜像的机制一起使用。使用许可证 URL 强制安装程序复制源镜像，然后再使用。
platform.gcp.osDisk.diskSizeGB	以 GB (GB) 为单位的磁盘大小。	16 GB 到 65536 GB 之间的任何大小。

参数	描述	值
platform.gcp.osDisk.diskType	磁盘类型。	默认 **pd-ssd** 或 **pd-standard** 磁盘类型。control plane 节点必须是 **pd-ssd** 磁盘类型。worker 节点可以是其中任一个类型。

4.6.5.2. GCP 的自定义 install-config.yaml 文件示例

您可以自定义 **install-config.yaml** 文件，以指定有关 OpenShift Container Platform 集群平台的更多信息，或修改所需参数的值。



重要

此示例 YAML 文件仅供参考。您必须使用安装程序来获取 **install-config.yaml** 文件，并且修改该文件。

```
apiVersion: v1
baseDomain: example.com (1)
controlPlane: (2) (3)
  hyperthreading: Enabled (4)
  name: master
  platform:
    gcp:
      type: n2-standard-4
      zones:
      - us-central1-a
      - us-central1-c
      osDisk:
        diskType: pd-ssd
        diskSizeGB: 1024
  replicas: 3
compute: (5) (6)
- hyperthreading: Enabled (7)
  name: worker
  platform:
    gcp:
      type: n2-standard-4
      zones:
      - us-central1-a
      - us-central1-c
      osDisk:
        diskType: pd-standard
        diskSizeGB: 128
  replicas: 3
metadata:
  name: test-cluster (8)
networking:
  clusterNetwork:
  - cidr: 10.128.0.0/14
    hostPrefix: 23
```

```
  machineNetwork:
  - cidr: 10.0.0.0/16
  networkType: OpenShiftSDN
  serviceNetwork:
  - 172.30.0.0/16
platform:
  gcp:
    projectID: openshift-production 9
    region: us-central1 10
    network: existing_vpc 11
    controlPlaneSubnet: control_plane_subnet 12
    computeSubnet: compute_subnet 13
pullSecret: '{"auths":{"<local_registry>": {"auth": "<credentials>","email": "you@example.com"}}}' 14
fips: false 15
sshKey: ssh-ed25519 AAAA... 16
additionalTrustBundle: | 17
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
imageContentSources: 18
- mirrors:
  - <local_registry>/<local_repository_name>/release
  source: quay.io/openshift-release-dev/ocp-release
- mirrors:
  - <local_registry>/<local_repository_name>/release
  source: quay.io/openshift-release-dev/ocp-v4.0-art-dev
```

1 8 9 10 必需。安装程序会提示您输入这个值。

2 5 如果没有提供这些参数和值，安装程序会提供默认值。

3 6 **controlPlane** 部分是一个单映射，但 **compute** 部分是一系列映射。为满足不同数据结构的要求，**compute** 部分的第一行必须以连字符 **-** 开头，**controlPlane** 部分的第一行则不可以连字符开头。只使用一个 control plane 池。

4 7 是否要启用或禁用并发多线程或**超线程**。默认情况下，启用并发多线程以提高机器内核的性能。您可以通过将参数值设为 **Disabled** 来禁用。如果您在某些集群机器上禁用并发多线程，则必须在所有集群机器上禁用。

重要

如果禁用并发多线程，请确保在容量规划时考虑到机器性能可能会显著降低的问题。如果您禁用并发多线程，请为您的机器使用较大的类型，如 **n1-standard-8**。

11 指定现有 VPC 的名称。

12 指定要将 control plane 机器部署到的现有子网的名称。该子网必须属于您指定的 VPC。

13 指定要将计算机器部署到的现有子网的名称。该子网必须属于您指定的 VPC。

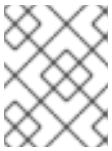
14 对于 **<local_registry>**，请指定 registry 域名，以及您的镜像 registry 用来提供内容的可选端口。例如： **registry.example.com** 或 **registry.example.com:5000**。使用 **<credentials>** 为您生成的镜像 registry 指定 base64 编码的用户名和密码。

15 是否启用或禁用 FIPS 模式。默认情况下不启用 FIPS 模式。如果启用了 FIPS 模式，运行 OpenShift Container Platform 的 Red Hat Enterprise Linux CoreOS (RHCOS) 机器会绕过默认的 Kubernetes

重要

只有在 **x86_64** 架构中的 OpenShift Container Platform 部署支持 FIPS 验证的/Modules in Process 加密库。

16 您可以选择提供您用来访问集群中机器的 **sshKey** 值。



注意

对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 **ssh-agent** 进程使用的 SSH 密钥。

17 提供用于镜像 registry 的证书文件内容。

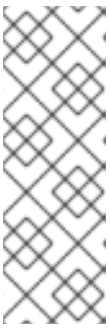
18 提供命令输出中的 **imageContentSources** 部分来镜像存储库。

4.6.5.3. 在安装过程中配置集群范围代理

生产环境可能会拒绝直接访问互联网，而是提供 HTTP 或 HTTPS 代理。您可以通过在 **install-config.yaml** 文件中配置代理设置，将新的 OpenShift Container Platform 集群配置为使用代理。

先决条件

- 您有一个现有的 **install-config.yaml** 文件。
- 您检查了集群需要访问的站点，并决定是否需要绕过代理。默认情况下代理所有集群出口流量，包括对托管云供应商 API 的调用。您需要将站点添加到 **Proxy** 对象的 **spec.noProxy** 字段来绕过代理。



注意

Proxy 对象 **status.noProxy** 字段使用安装配置中的 **networking.machineNetwork[].cidr**、**networking.clusterNetwork[].cidr** 和 **networking.serviceNetwork[]** 字段的值填充。

对于在 Amazon Web Services(AWS)、Google Cloud Platform(GCP)、Microsoft Azure 和 Red Hat OpenStack Platform(RHOSP)上安装, **Proxy** 对象 **status.noProxy** 字段也会使用实例元数据端点填充(**169.254.169.254**)。

流程

1. 编辑 **install-config.yaml** 文件并添加代理设置。例如：

```
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: example.com 3
```

```
additionalTrustBundle: | 4
  -----BEGIN CERTIFICATE-----
  <MY_TRUSTED_CA_CERT>
  -----END CERTIFICATE-----
...
```

1. 用于创建集群外 HTTP 连接的代理 URL。URL 必须是 **http**。
2. 用于创建集群外 HTTPS 连接的代理 URL。
3. 要排除在代理中的目标域名、IP 地址或其他网络 CIDR 的逗号分隔列表。在域前面加 **.** 来仅匹配子域。例如： **.y.com** 匹配 **x.y.com**，但不匹配 **y.com**。使用 * 绕过所有目的地的代理。
4. 如果提供，安装程序会在 **openshift-config 命名空间中生成名为 user-ca- bundle** 的配置映射来保存额外的 CA 证书。如果您提供 **additionalTrustBundle** 和至少一个代理设置，则 **Proxy** 对象会被配置为引用 **trustedCA** 字段中的 **user-ca-bundle** 配置映射。然后，Cluster Network Operator 会创建一个 **trusted-ca-bundle** 配置映射，该配置映射将为 **trustedCA** 参数指定的内容与 RHCOS 信任捆绑包合并。 **additionalTrustBundle** 字段是必需的，除非代理的身份证书由来自 RHCOS 信任捆绑包的颁发机构签名。

注意

安装程序不支持代理的 **readinessEndpoints** 字段。

2. 保存该文件，并在安装 OpenShift Container Platform 时引用。

安装程序会创建一个名为 **cluster** 的集群范围代理，该代理使用提供的 **install-config.yaml** 文件中的代理设置。如果没有提供代理设置，仍然会创建一个 **cluster Proxy** 对象，但它会有一个空 **spec**。

注意

只支持名为 **cluster** 的 **Proxy** 对象，且无法创建额外的代理。

4.6.6. 部署集群

您可以在兼容云平台中安装 OpenShift Container Platform。

重要

安装程序的 **create cluster** 命令只能在初始安装过程中运行一次。

先决条件

- 配置托管集群的云平台的帐户。
- 获取 OpenShift Container Platform 安装程序以及集群的 pull secret。

流程

1. 对于存储在以下位置的任何现有 GCP 凭证，删除不使用您为集群配置的 GCP 帐户的服务帐户密钥的凭证：

- **GOOGLE_CREDENTIALS**、**GOOGLE_CLOUD_KEYFILE_JSON** 或 **GCLOUD_KEYFILE_JSON** 环境变量
- **~/.gcp/osServiceAccount.json** 文件
- **gcloud cli** 默认凭证

2. 更改为包含安装程序的目录并初始化集群部署：

```
$ ./openshift-install create cluster --dir <installation_directory> \ 1
    --log-level=info 2
```

1 对于 **<installation_directory>**，请指定自定义 **./install-config.yaml** 文件的位置。

2 要查看不同的安装详情，请指定 **warn**、**debug** 或 **error**，而不要指定 **info**。

注意

如果您在主机上配置的云供应商帐户没有足够的权限来部署集群，安装过程将会停止，并且显示缺少的权限。

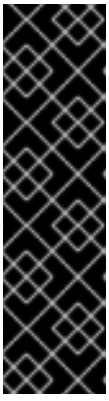
集群部署完成后，终端会显示访问集群的信息，包括指向其 Web 控制台的链接和 **kubeadmin** 用户的凭证。

输出示例

```
...
INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run 'export
KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'
INFO Access the OpenShift web-console here: https://console-openshift-
console.apps.mycluster.example.com
INFO Login to the console with user: "kubeadmin", and password: "4vYBz-Ee6gm-ymBZj-
Wt5AL"
INFO Time elapsed: 36m22s
```

注意

当安装成功时，集群访问和凭证信息还会输出到 **<installation_directory>/.openshift_install.log**。



重要

- 安装程序生成的 Ignition 配置文件包含在 24 小时后过期的证书，然后在过期时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外情况是，您需要手动批准待处理的 **node-bootstrapper** 证书签名请求（CSR）来恢复 kubelet 证书。如需更多信息，请参阅*从过期的 control plane 证书中恢复*的文档。
- 建议您在生成 12 小时后使用 Ignition 配置文件，因为集群安装后 24 小时证书从 16 小时轮转至 22 小时。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中运行证书更新时避免安装失败。

重要

您不得删除安装程序或安装程序所创建的文件。需要这两者才能删除集群。

3. 可选：您可以减少用于安装集群的服务帐户的权限数。
   - 如果将 **Owner** 角色分配给您的服务帐户，您可以删除该角色并使用 **Viewer** 角色替换。
   - 如果您包含 **Service Account Key Admin** 角色，您可以将其删除。

4.6.7. 通过下载二进制文件安装 OpenShift CLI

您需要安装 CLI（**oc**） 来使用命令行界面与 OpenShift Container Platform 进行交互。您可在 Linux 、Windows 或 macOS 上安装 **oc**。

重要

如果安装了旧版本的 **oc**，则无法使用 OpenShift Container Platform 4.6 中的所有命令。下载并安装新版本的 **oc**。

4.6.7.1. 在 Linux 上安装 OpenShift CLI

您可以按照以下流程在 Linux 上安装 OpenShift CLI（**oc**）二进制文件。

流程

1. 进入到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 在 **Version** 下拉菜单中选择相应的版本。
3. 单击 **OpenShift v4.6 Linux 客户端** 条目旁边的 **Download Now**，再保存文件。
4. 解包存档：

```
$ tar xvzf <file>
```

5. 把 **oc** 二进制代码放到 **PATH** 中的目录中。
   执行以下命令可以查看当前的 **PATH** 设置：

```
$ echo $PATH
```

安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
$ oc <command>
```

4.6.7.2. 在 Windows 上安装 OpenShift CLI

您可以按照以下流程在 Windows 上安装 OpenShift CLI（**oc**）二进制代码。

流程

1. 进入到红帽客户门户网站上的 OpenShift Container Platform 下载页面。

2. 在 **Version** 下拉菜单中选择相应的版本。
3. 单击 **OpenShift v4.6 Windows 客户端**条目旁边的 **Download Now**，再保存文件。
4. 使用 ZIP 程序解压存档。
5. 把 **oc** 二进制代码放到 **PATH** 中的目录中。
   要查看您的 **PATH**，请打开命令提示窗口并执行以下命令：

```
C:\> path
```

安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
C:\> oc <command>
```

4.6.7.3. 在 macOS 上安装 OpenShift CLI

您可以按照以下流程在 macOS 上安装 OpenShift CLI（**oc**）二进制代码。

流程

1. 进入到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 在 **Version** 下拉菜单中选择相应的版本。
3. 单击 **OpenShift v4.6 MacOSX 客户端**条目旁边的 **Download Now**，再保存文件。
4. 解包和解压存档。
5. 将 **oc** 二进制文件移到 PATH 的目录中。
   要查看您的 **PATH**，打开一个终端窗口并执行以下命令：

```
$ echo $PATH
```

安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
$ oc <command>
```

4.6.8. 使用 CLI 登录到集群

您可以通过导出集群 **kubeconfig** 文件，以默认系统用户身份登录集群。**kubeconfig** 文件包含关于集群的信息，供 CLI 用于将客户端连接到正确集群和 API 服务器。该文件特只适用于一个特定的集群，在 OpenShift Container Platform 安装过程中创建。

先决条件

- 已部署了 OpenShift Container Platform 集群。
- 已安装 **oc** CLI。

流程

1. 导出 **kubeadmin** 凭证：

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

2. 使用导出的配置，验证能否成功运行 **oc** 命令：

```
$ oc whoami
```

输出示例

```
system:admin
```

4.6.9. 禁用默认的 OperatorHub 源

在 OpenShift Container Platform 安装过程中，默认为 OperatorHub 配置由红帽和社区项目提供的源内容的 operator 目录。在受限网络环境中，必须以集群管理员身份禁用默认目录。

流程

- 通过在 **OperatorHub** 对象中添加 **disableAllDefaultSources: true** 来禁用默认目录的源：

```
$ oc patch OperatorHub cluster --type json \
    -p '[{"op": "add", "path": "/spec/disableAllDefaultSources", "value": true}]'
```

提示

或者，您可以使用 Web 控制台管理目录源。在 **Administration → Cluster Settings → Global Configuration → OperatorHub** 页面中，点 **Sources** 选项卡，其中可创建、删除、禁用和启用单独的源。

4.6.10. OpenShift Container Platform 的 Telemetry 访问

在 OpenShift Container Platform 4.6 中，默认运行的 Telemetry 服务提供有关集群健康状况和成功更新的指标，需要访问互联网。如果您的集群连接到互联网，Telemetry 会自动运行，而且集群会注册到 OpenShift Cluster Manager。

确认 OpenShift Cluster Manager 清单正确后，可以由 Telemetry 自动维护，也可以使用 OpenShift Cluster Manager 手动维护，使用订阅监控来跟踪帐户或多集群级别的 OpenShift Container Platform 订阅。

其他资源

- 有关 Telemetry 服务的更多信息，请参阅关于远程健康监控。

4.6.11. 后续步骤

- 验证安装。
- 自定义集群。
- 为 Cluster Samples Operator 和 **must-gather** 工具配置镜像流。

- 了解如何在受限网络中使用 Operator Lifecycle Manager（OLM）。
- 如果您用来安装集群的镜像 registry 具有一个可信任的 CA，通过配置额外的信任存储将其添加到集群中。
- 如果需要，您可以选择不使用远程健康报告。

4.7. 将 GCP 上的集群安装到现有的 VPC 中

在 OpenShift Container Platform 版本 4.6 中，您可以在 Google Cloud Platform（GCP）上将集群安装到现有的 Virtual Private Cloud（VPC）中。安装程序会置备所需基础架构的其余部分，您可以进一步定制这些基础架构。要自定义安装，请在安装集群前修改 **install-config.yaml** 文件中的参数。

4.7.1. 先决条件

- 查看有关 OpenShift Container Platform 安装和更新流程的详细信息。
- 配置 GCP 帐户以托管集群。
- 如果使用防火墙，则必须将其配置为允许集群需要访问的站点。
- 如果不允许系统管理身份和访问管理（IAM），集群管理员可以 手动创建和维护 IAM 凭证。手动模式也可以用于云 IAM API 无法访问的环境中。

4.7.2. 关于使用自定义 VPC

在 OpenShift Container Platform 4.6 中，您可以在 Google Cloud Platform(GCP)的现有 Virtual Private Cloud(VPC)中将集群部署到现有子网中。通过将 OpenShift Container Platform 部署到现有的 GCP VPC 中，您可能会避开新帐户中的限制，或者更容易地利用公司所设置的操作限制。如果您无法获得您自己创建 VPC 所需的基础架构创建权限，请使用这个安装选项。您必须为子网配置网络。

4.7.2.1. 使用 VPC 的要求

VPC CIDR 块的合并，机器网络 CIDR 必须不是空的。子网必须位于机器网络中。

安装程序不会创建以下组件：

- NAT 网关
- 子网
- 路由表
- VPC 网络

注意

安装程序要求您使用由云提供的 DNS 服务器。不支持使用自定义 DNS 服务器，并导致安装失败。

4.7.2.2. VPC 验证

要确保您提供的子网适合您的环境，安装程序会确认以下信息：

- 您指定的所有子网都存在。

- 您可以为 control-plane 机器和一个子网提供计算机器。
- 子网的 CIDR 属于您指定的机器 CIDR。

4.7.2.3. 权限划分

有些个人可以在您的云中创建不同的资源。例如，您可以创建针对于特定应用程序的对象，如实例、存储桶和负载均衡器，但不能创建与网络相关的组件，如 VPC 、子网或入站规则。

4.7.2.4. 集群间隔离

如果您将 OpenShift Container Platform 部署到现有网络中，集群服务的隔离将在以下方面减少：

- 您可以在同一 VPC 中安装多个 OpenShift Container Platform 集群。
- 整个网络都允许 ICMP 入站流量。
- 整个网络都允许 TCP 22 入站流量 (SSH)。
- 整个网络都允许 control plane TCP 6443 入站流量 (Kubernetes API)。
- 整个网络都允许 control plane TCP 22623 入站流量 (MCS) 。

4.7.3. OpenShift Container Platform 的互联网访问

在 OpenShift Container Platform 4.6 中，您需要访问互联网来安装集群。

您必须具有以下互联网访问权限：

- 访问 OpenShift Cluster Manager 以下载安装程序并执行订阅管理。如果集群可以访问互联网，并且没有禁用 Telemetry，该服务会自动授权您的集群。
- 访问 Quay.io，以获取安装集群所需的软件包。
- 获取执行集群更新所需的软件包。

重要

如果您的集群无法直接访问互联网，则可以在置备的某些类基础架构上执行受限网络安装。在此过程中，您要下载所需的内容，并使用它在镜像 registry（mirror registry） 中填充安装集群并生成安装程序所需的软件包。对于某些安装类型，集群要安装到的环境不需要访问互联网。在更新集群之前，要更新 registry 镜像系统中的内容。

4.7.4. 生成 SSH 私钥并将其添加到代理中

如果要在集群上执行安装调试或灾难恢复，则必须为 **ssh-agent** 和安装程序提供 SSH 密钥。您可以使用此密钥访问公共集群中的 bootstrap 机器来排除安装问题。

注意

在生产环境中，您需要进行灾难恢复和调试。

您可以使用此密钥以 **core** 用户身份通过 SSH 连接到 master 节点。在部署集群时，此密钥会添加到 **core** 用户的 **~/.ssh/authorized_keys** 列表中。

注意

您必须使用一个本地密钥，而不要使用在特定平台上配置的密钥，如 AWS 密钥对。

流程

1. 如果还没有为计算机上免密码身份验证而配置的 SSH 密钥，请创建一个。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ ssh-keygen -t ed25519 -N '' \
  -f <path>/<file_name> 1
```

1 指定新 SSH 密钥的路径和文件名，如 **~/.ssh/id_rsa**。如果您已有密钥对，请确保您的公钥位于 **~/.ssh** 目录中。

运行此命令会在指定的位置生成不需要密码的 SSH 密钥。

注意

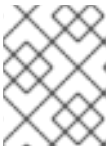
如果您计划在 **x86_64** 架构中安装使用 FIPS 验证的/Modules in Process 加密库的 OpenShift Container Platform 集群，不要创建使用 **ed25519** 算法的密钥。反之，创建一个使用 **rsa** 或 **ecdsa** 算法的密钥。

2. 作为后台任务启动 **ssh-agent** 进程：

```
$ eval "$(ssh-agent -s)"
```

输出示例

```
Agent pid 31874
```



注意

如果您的集群采用 FIPS 模式，则只使用 FIPS 兼容算法来生成 SSH 密钥。密钥必须是 RSA 或 ECDSA。

3. 将 SSH 私钥添加到 **ssh-agent**：

```
$ ssh-add <path>/<file_name> 1
```

输出示例

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

1 指定 SSH 私钥的路径和文件名，如 **~/.ssh/id_rsa**

4. 将 **GOOGLE_APPLICATION_CREDENTIALS** 环境变量设置为服务帐户私钥文件的完整路径。

```
$ export GOOGLE_APPLICATION_CREDENTIALS="<your_service_account_file>"
```

5. 验证是否应用了凭证。

```
$ gcloud auth list
```

后续步骤

- 在安装 OpenShift Container Platform 时，为安装程序提供 SSH 公钥。

4.7.5. 获取安装程序

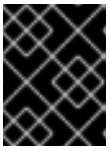
在安装 OpenShift Container Platform 之前，将安装文件下载到本地计算机上。

先决条件

- 运行 Linux 或 macOS 的计算机，本地磁盘空间为 500 MB

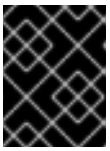
流程

1. 访问 OpenShift Cluster Manager 站点的 Infrastructure Provider 页面。如果您有红帽帐号，请使用自己的凭证登录。如果没有，请创建一个帐户。
2. 选择您的基础架构供应商。
3. 进入适用于您的安装类型的页面，下载您的操作系统的安装程序，并将文件放在要保存安装配置文件的目录中。。



重要

安装程序会在用来安装集群的计算机上创建若干文件。在完成集群安装后，您必须保留安装程序和安装程序所创建的文件。这两个文件都需要删除集群。



重要

删除安装程序创建的文件不会删除您的集群，即使集群在安装过程中失败也是如此。要删除集群，为特定云供应商完成 OpenShift Container Platform 卸载流程。

4. 提取安装程序。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ tar xvf openshift-install-linux.tar.gz
```

5. 从 Red Hat OpenShift Cluster Manager 下载安装 pull secret 。通过此 pull secret，您可以进行所含授权机构提供的服务的身份验证，这些服务包括为 OpenShift Container Platform 组件提供容器镜像的 Quay.io。

4.7.6. 创建安装配置文件

您可以自定义在 Google Cloud Platform (GCP) 上安装的 OpenShift Container Platform 集群。

先决条件

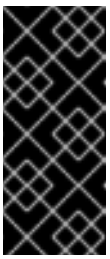
- 获取 OpenShift Container Platform 安装程序以及集群的 pull secret。

流程

1. 创建 **install-config.yaml** 文件。
   a. 更改到包含安装程序的目录，再运行以下命令：

```
$ ./openshift-install create install-config --dir <installation_directory> 1
```

   1 对于 **<installation_directory>**，请指定用于保存安装程序所创建的文件的目录名称。



重要

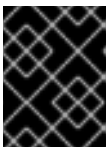
指定一个空目录。一些安装信息，如 bootstrap X.509 证书，有较短的过期间隔，因此不要重复使用安装目录。如果要重复使用另一个集群安装中的个别文件，可以将其复制到您的目录中。但是，一些安装数据的文件名可能会在发行版本之间有所改变。从 OpenShift Container Platform 老版本中复制安装文件时要格外小心。

   b. 在提示符处，提供您的云的配置详情：
      i. 可选：选择用来访问集群机器的 SSH 密钥。

注意

对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 **ssh-agent** 进程使用的 SSH 密钥。

      ii. 选择 **gcp** 作为目标平台。
      iii. 如果您没有为计算机上的 GCP 帐户配置服务帐户密钥，则必须从 GCP 获取，并粘贴文件的内容或输入文件的绝对路径。
      iv. 选择要在其中置备集群的项目 ID。默认值由您配置的服务帐户指定。
      v. 选择要在其中部署集群的区域。
      vi. 选择集群要部署到的基域。基域与您为集群创建的公共 DNS 区对应。
      vii. 为集群输入一个描述性名称。
      viii. 粘贴 Red Hat OpenShift Cluster Manager 中的 pull secret 。
2. 修改 **install-config.yaml** 文件。您可以在**安装配置参数**部分中找到有关可用参数的更多信息。
3. 备份 **install-config.yaml** 文件，以便用于安装多个集群。

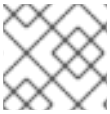


重要

install-config.yaml 文件会在安装过程中消耗掉。如果要重复使用此文件，必须现在备份。

4.7.6.1. 安装配置参数

在部署 OpenShift Container Platform 集群前，您可以提供参数值，以描述托管集群的云平台的帐户并选择性地自定义集群平台。在创建 **install-config.yaml** 安装配置文件时，您可以通过命令行来提供所需的参数的值。如果要自定义集群，可以修改 **install-config.yaml** 文件来提供关于平台的更多信息。



注意

安装之后，您无法修改 **install-config.yaml** 文件中的这些参数。

重要

openshift-install 命令不验证参数的字段名称。如果指定了不正确的名称，则不会创建相关的文件或对象，且不会报告错误。确保所有指定的参数的字段名称都正确。

4.7.6.1.1. 所需的配置参数

下表描述了所需的安装配置参数：

表 4.21. 所需的参数

参数	描述	值
apiVersion	**install-config.yaml** 内容的 API 版本。当前版本是 **v1**。安装程序还可能支持旧的 API 版本。	字符串
baseDomain	云供应商的基域。此基础域用于创建到 OpenShift Container Platform 集群组件的路由。集群的完整 DNS 名称是 **baseDomain** 和 **metadata.name** 参数值的组合，其格式为 **<metadata.name>.<baseDomain>**。	完全限定域名或子域名，如 **example.com**。
metadata	Kubernetes 资源 **ObjectMeta**，其中只消耗 **name** 参数。	对象
metadata.name	集群的名称。集群的 DNS 记录是 **{{.metadata.name}}.{{.baseDomain}}** 的子域。	小写字母,连字符(-)和句点(.)的字符串，如 **dev**。
platform	执行安装的具体平台配置：**aws**、**baremetal**、**azure**、**openstack**、**ovirt**、**vsphere**。有关 **platform.<platform>** 参数的额外信息，请参考下表来了解您的具体平台。	对象

参数	描述	值

pullSecret	从 Red Hat OpenShift Cluster Manager 获取 pull secret，验证从 Quay.io 等服务中下载 OpenShift Container Platform 组件的容器镜像。	<pre>{
		  "auths":{
		  "cloud.openshift.com":{
		  "auth":"b3Blb=",
		  "email":"you@example.com"
		  },
		  "quay.io":{
		  "auth":"b3Blb=",
		  "email":"you@example.com"
		  }
		  }
		}</pre>

4.7.6.1.2. 网络配置参数

您可以根据现有网络基础架构的要求自定义安装配置。例如，您可以扩展集群网络的 IP 地址块，或者提供不同于默认值的不同 IP 地址块。

只支持 IPv4 地址。

表 4.22. 网络参数

参数	描述	值
networking	集群网络的配置。	对象 **注意** 您不能在安装后修改 **networking** 对象指定的参数。
networking.networkType	要安装的集群网络供应商 Container Network Interface（CNI）插件。	**OpenShiftSDN** 或 **OVNKubernetes**。默认值为 **OpenShiftSDN**。

networking.clusterNetwork	pod 的 IP 地址块。
	
	默认值为 **10.128.0.0/14**，主机前缀为 **/23**。
	
	如果您指定多个 IP 地址块，则块不得互相重叠。
		一个对象数组。例如：
		
		<pre>networking:
		  clusterNetwork:
		  - cidr: 10.128.0.0/14
		  hostPrefix: 23</pre>

networking.clusterNetwork.cidr	使用 **networking.clusterNetwork** 时需要此项。IP 地址块。 一个 IPv4 网络。	使用 CIDR 形式的 IP 地址块。IPv4 块的前缀长度介于 **0** 到 **32** 之间。

参数	描述	值
networking.clusterNetwork.hostPrefix	分配给每个单独节点的子网前缀长度。例如，如果 **hostPrefix** 设为 **23**，则每个节点从所给的 **cidr** 中分配一个 **/23** 子网。**hostPrefix** 值 **23** 提供 510（2^(32 - 23)- 2）个 pod IP 地址。	子网前缀。 默认值为 **23**。
networking.serviceNetwork	服务的 IP 地址块。默认值为 **172.30.0.0/16**。 OpenShift SDN 和 OVN-Kubernetes 网络供应商只支持服务网络的一个 IP 地址块。	CIDR 格式具有 IP 地址块的数组。例如： networking: serviceNetwork: - 172.30.0.0/16
networking.machineNetwork	机器的 IP 地址块。 如果您指定多个 IP 地址块，则块不得互相重叠。	一个对象数组。例如： networking: machineNetwork: - cidr: 10.0.0.0/16
networking.machineNetwork.cidr	使用 **networking.machineNetwork** 时需要。IP 地址块。libvirt 以外的所有平台的默认值为 **10.0.0.0/16**。对于 libvirt，默认值为 **192.168.126.0/24**。	CIDR 表示法中的 IP 网络块。 例如： **10.0.0.0/16**。 **注意** 将 **networking.machineNetwork** 设置为与首选 NIC 所在的 CIDR 匹配。

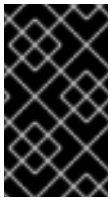
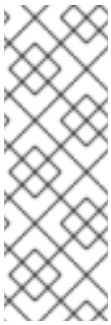
4.7.6.1.3. 可选配置参数

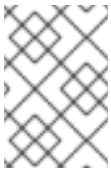
下表描述了可选安装配置参数：

表 4.23. 可选参数

参数	描述	值
additionalTrustBundle	添加到节点可信证书存储中的 PEM 编码 X.509 证书捆绑包。配置了代理时，也可以使用这个信任捆绑包。	字符串

参数	描述	值
compute	组成计算节点的机器的配置。	machine-pool 对象的数组。详情请查看以下"Machine-pool"表。
compute.architecture	决定池中机器的指令集合架构。目前不支持异构集群，因此所有池都必须指定相同的架构。有效值为 **amd64** （默认值）。	字符串
compute.hyperthreading	是否在计算机器上启用或禁用并发多线程或**超线程**。默认情况下，启用并发多线程以提高机器内核的性能。  **重要** 如果禁用并发多线程，请确保在容量规划时考虑到机器性能可能会显著降低的问题。	**Enabled** 或 **Disabled**
compute.name	使用 **compute** 时需要此值。机器池的名称。	**worker**
compute.platform	使用 **compute** 时需要此值。使用此参数指定托管 worker 机器的云供应商。此参数值必须与 **controlPlane.platform** 参数值匹配。	**aws**、**azure**、**gcp**、**openstack**、**ovirt**、**vsphere** 或 **{}**
compute.replicas	要置备的计算机器数量，也称为 worker 机器。	大于或等于 **2** 的正整数。默认值为 **3**。
controlPlane	组成 control plane 的机器的配置。	**MachinePool** 对象的数组。详情请查看以下"Machine-pool"表。
controlPlane.architecture	决定池中机器的指令集合架构。目前不支持异构集群，因此所有池都必须指定相同的架构。有效值为 **amd64** （默认值）。	字符串

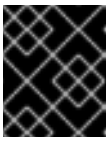
参数	描述	值
controlPlane.hyperthreading	是否在 control plane 机器上启用或禁用并发多线程或**超线程**。默认情况下，启用并发多线程以提高机器内核的性能。  **重要** 如果禁用并发多线程，请确保在容量规划时考虑到机器性能可能会显著降低的问题。	**Enabled** 或 **Disabled**
controlPlane.name	使用 **controlPlane** 时需要。机器池的名称。	**master**
controlPlane.platform	使用 **controlPlane** 时需要。使用此参数指定托管 control plane 机器的云供应商。此参数值必须与 **compute.platform** 参数值匹配。	**aws**、**azure**、**gcp**、**openstack**、**ovirt**、**vsphere** 或 **{}**
controlPlane.replicas	要置备的 control plane 机器数量。	唯一支持的值是 **3**，它是默认值。
credentialsMode	Cloud Credential Operator（CCO）模式。如果没有指定任何模式，CCO 会动态地尝试决定提供的凭证的功能，在支持多个模式的平台上使用 mint 模式。  **注意** 不是所有 CCO 模式都支持所有云供应商。如需有关 CCO 模式的更多信息，请参阅*Red Hat Operator 参考指南*内容中的 *Cloud Credential Operator* 条目。	**Mint**、**Passthrough**、**Manual** 或空字符串(**""**)。

参数	描述	值
fips	启用或禁用 FIPS 模式。默认为 **false**（禁用）。如果启用了 FIPS 模式，运行 OpenShift Container Platform 的 Red Hat Enterprise Linux CoreOS (RHCOS) 机器会绕过默认的 Kubernetes 加密套件，并使用由 RHCOS 提供的加密模块。  **重要** 只有在 **x86_64** 架构中的 OpenShift Container Platform 部署支持 FIPS 验证的/Modules in Process 加密库。  **注意** 如果使用 Azure File 存储，则无法启用 FIPS 模式。	**false** 或 **true**
imageContentSources	release-image 内容的源和仓库。	对象数组。包括一个 **source** 以及可选的 **mirrors**，如下表所示。
imageContentSources.source	使用 **imageContentSources** 时需要。指定用户在镜像拉取规格中引用的仓库。	字符串
imageContentSources.mirrors	指定可能还包含同一镜像的一个或多个仓库。	字符串数组
publish	如何发布或公开集群的面向用户的端点，如 Kubernetes API、OpenShift 路由。	**Internal** 或 **External**。把 **publish** 设置为 **Internal** 以部署一个私有集群，它不能被互联网访问。默认值为 **External**。
sshKey	用于验证集群机器访问的 SSH 密钥或密钥。  **注意** 对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 **ssh-agent** 进程使用的 SSH 密钥。	一个或多个密钥。例如： sshKey: <key1> <key2> <key3>

4.7.6.1.4. 其他 Google Cloud Platform（GCP）配置参数

下表中描述了额外的 GCP 配置参数：

表 4.24. 额外的 GCP 参数

参数	描述	值
platform.gcp.network	要将集群部署到的现有 VPC 的名称。	字符串。
platform.gcp.region	托管集群的 GCP 区域的名称。	任何有效的区域名称，如 **us-central1**。
platform.gcp.type	GCP 机器类型。	GCP 机器类型。
platform.gcp.zones	安装程序在其中为特定 MachinePool 创建机器的可用区。	有效的 GCP 可用区 列表，如 **us-central1-a**，在一个 YAML 序列中。
platform.gcp.controlPlaneSubnet	要将 control plane 机器部署到的 VPC 中现有子网的名称。	子网名称。
platform.gcp.computeSubnet	您要将计算机器部署到的 VPC 中现有子网的名称。	子网名称。
platform.gcp.licenses	必须应用到计算镜像的许可证 URL 列表。  **重要** License **参数** 是一个已弃用的字段，嵌套虚拟化默认为启用。不建议使用此字段。	许可证 API 可获得的任何许可证，如启用 嵌套虚拟化 的许可证。您不能将此参数与生成预构建镜像的机制一起使用。使用许可证 URL 强制安装程序复制源镜像，然后再使用。
platform.gcp.osDisk.diskSizeGB	以 GB (GB) 为单位的磁盘大小。	16 GB 到 65536 GB 之间的任何大小。

参数	描述	值
platform.gcp.osDisk.diskType	磁盘类型。	默认 **pd-ssd** 或 **pd-standard** 磁盘类型。control plane 节点必须是 **pd-ssd** 磁盘类型。worker 节点可以是其中任一个类型。

4.7.6.2. GCP 的自定义 install-config.yaml 文件示例

您可以自定义 **install-config.yaml** 文件，以指定有关 OpenShift Container Platform 集群平台的更多信息，或修改所需参数的值。

重要

此示例 YAML 文件仅供参考。您必须使用安装程序来获取 **install-config.yaml** 文件，并且修改该文件。

```
apiVersion: v1
baseDomain: example.com (1)
controlPlane: (2) (3)
  hyperthreading: Enabled (4)
  name: master
  platform:
    gcp:
      type: n2-standard-4
      zones:
      - us-central1-a
      - us-central1-c
      osDisk:
        diskType: pd-ssd
        diskSizeGB: 1024
  replicas: 3
compute: (5) (6)
- hyperthreading: Enabled (7)
  name: worker
  platform:
    gcp:
      type: n2-standard-4
      zones:
      - us-central1-a
      - us-central1-c
      osDisk:
        diskType: pd-standard
        diskSizeGB: 128
  replicas: 3
metadata:
  name: test-cluster (8)
networking:
  clusterNetwork:
  - cidr: 10.128.0.0/14
    hostPrefix: 23
```

```
  machineNetwork:
  - cidr: 10.0.0.0/16
  networkType: OpenShiftSDN
  serviceNetwork:
  - 172.30.0.0/16
platform:
  gcp:
    projectID: openshift-production 9
    region: us-central1 10
    network: existing_vpc 11
    controlPlaneSubnet: control_plane_subnet 12
    computeSubnet: compute_subnet 13
pullSecret: '{"auths": ...}' 14
fips: false 15
sshKey: ssh-ed25519 AAAA... 16
```

1 8 9 10 14 必需。安装程序会提示您输入这个值。

2 5 如果没有提供这些参数和值，安装程序会提供默认值。

3 6 **controlPlane** 部分是一个单映射，但 **compute** 部分是一系列映射。为满足不同数据结构的要求，**compute** 部分的第一行必须以连字符 **-** 开头，**controlPlane** 部分的第一行则不可以连字符开头。只使用一个 control plane 池。

4 7 是否要启用或禁用并发多线程或**超线程**。默认情况下，启用并发多线程以提高机器内核的性能。您可以通过将参数值设为 **Disabled** 来禁用。如果您在某些集群机器上禁用并发多线程，则必须在所有集群机器上禁用。

重要

如果禁用并发多线程，请确保在容量规划时考虑到机器性能可能会显著降低的问题。如果您禁用并发多线程，请为您的机器使用较大的类型，如 **n1-standard-8**。

11 指定现有 VPC 的名称。

12 指定要将 control plane 机器部署到的现有子网的名称。该子网必须属于您指定的 VPC。

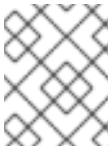
13 指定要将计算机器部署到的现有子网的名称。该子网必须属于您指定的 VPC。

15 是否启用或禁用 FIPS 模式。默认情况下不启用 FIPS 模式。如果启用了 FIPS 模式，运行 OpenShift Container Platform 的 Red Hat Enterprise Linux CoreOS (RHCOS) 机器会绕过默认的 Kubernetes 加密套件，并使用由 RHCOS 提供的加密模块。

重要

只有在 **x86_64** 架构中的 OpenShift Container Platform 部署支持 FIPS 验证的/Modules in Process 加密库。

16 您可以选择提供您用来访问集群中机器的 **sshKey** 值。



注意

对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 **ssh-agent** 进程使用的 SSH 密钥。

4.7.6.3. 在安装过程中配置集群范围代理

生产环境可能会拒绝直接访问互联网，而是提供 HTTP 或 HTTPS 代理。您可以通过在 **install-config.yaml** 文件中配置代理设置，将新的 OpenShift Container Platform 集群配置为使用代理。

先决条件

- 您有一个现有的 **install-config.yaml** 文件。
- 您检查了集群需要访问的站点，并决定是否需要绕过代理。默认情况下代理所有集群出口流量，包括对托管云供应商 API 的调用。您需要将站点添加到 **Proxy** 对象的 **spec.noProxy** 字段来绕过代理。

注意

Proxy 对象 **status.noProxy** 字段使用安装配置中的 **networking.machineNetwork[].cidr**、**networking.clusterNetwork[].cidr** 和 **networking.serviceNetwork[]** 字段的值填充。

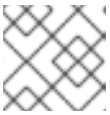
对于在 Amazon Web Services(AWS)、Google Cloud Platform(GCP)、Microsoft Azure 和 Red Hat OpenStack Platform(RHOSP)上安装, **Proxy** 对象 **status.noProxy** 字段也会使用实例元数据端点填充(**169.254.169.254**)。

流程

1. 编辑 **install-config.yaml** 文件并添加代理设置。例如：

```
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: example.com 3
additionalTrustBundle: | 4
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
...
```

1. 用于创建集群外 HTTP 连接的代理 URL。URL 必须是 **http**。
2. 用于创建集群外 HTTPS 连接的代理 URL。
3. 要排除在代理中的目标域名、IP 地址或其他网络 CIDR 的逗号分隔列表。在域前面加 **.** 来仅匹配子域。例如： **.y.com** 匹配 **x.y.com**，但不匹配 **y.com**。使用 ***** 绕过所有目的地的代理。
4. 如果提供，安装程序会在 **openshift-config 命名空间中生成名为 user-ca- bundle** 的配置映

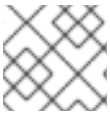


注意

安装程序不支持代理的 **readinessEndpoints** 字段。

2. 保存该文件，并在安装 OpenShift Container Platform 时引用。

安装程序会创建一个名为 **cluster** 的集群范围代理，该代理使用提供的 **install-config.yaml** 文件中的代理设置。如果没有提供代理设置，仍然会创建一个 **cluster Proxy** 对象，但它会有一个空 **spec**。



注意

只支持名为 **cluster** 的 **Proxy** 对象，且无法创建额外的代理。

4.7.7. 部署集群

您可以在兼容云平台中安装 OpenShift Container Platform。

重要

安装程序的 **create cluster** 命令只能在初始安装过程中运行一次。

先决条件

- 配置托管集群的云平台的帐户。
- 获取 OpenShift Container Platform 安装程序以及集群的 pull secret。

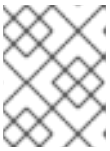
流程

1. 对于存储在以下位置的任何现有 GCP 凭证，删除不使用您为集群配置的 GCP 帐户的服务帐户密钥的凭证：
   - **GOOGLE_CREDENTIALS**、**GOOGLE_CLOUD_KEYFILE_JSON** 或 **GCLOUD_KEYFILE_JSON** 环境变量
   - **~/.gcp/osServiceAccount.json** 文件
   - **gcloud cli** 默认凭证
2. 更改为包含安装程序的目录并初始化集群部署：

```
$ ./openshift-install create cluster --dir <installation_directory> \ 1
    --log-level=info 2
```

1 对于 **<installation_directory>**，请指定自定义 **./install-config.yaml** 文件的位置。

2 要查看不同的安装详情，请指定 **warn**、**debug** 或 **error**，而不要指定 **info**。



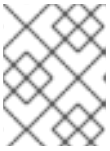
注意

如果您在主机上配置的云供应商帐户没有足够的权限来部署集群，安装过程将会停止，并且显示缺少的权限。

集群部署完成后，终端会显示访问集群的信息，包括指向其 Web 控制台的链接和 **kubeadmin** 用户的凭证。

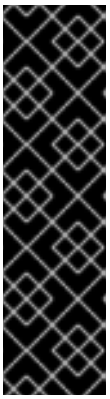
输出示例

```
...
INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run 'export
KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'
INFO Access the OpenShift web-console here: https://console-openshift-
console.apps.mycluster.example.com
INFO Login to the console with user: "kubeadmin", and password: "4vYBz-Ee6gm-ymBZj-
Wt5AL"
INFO Time elapsed: 36m22s
```



注意

当安装成功时，集群访问和凭证信息还会输出到 **<installation_directory>/.openshift_install.log**。



重要

- 安装程序生成的 Ignition 配置文件包含在 24 小时后过期的证书，然后在过期时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外情况是，您需要手动批准待处理的 **node-bootstrapper** 证书签名请求（CSR）来恢复 kubelet 证书。如需更多信息，请参阅*从过期的 control plane 证书中恢复*的文档。
- 建议您在生成 12 小时后使用 Ignition 配置文件，因为集群安装后 24 小时证书从 16 小时轮转至 22 小时。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中运行证书更新时避免安装失败。

重要

您不得删除安装程序或安装程序所创建的文件。需要这两者才能删除集群。

3. 可选：您可以减少用于安装集群的服务帐户的权限数。
   - 如果将 **Owner** 角色分配给您的服务帐户，您可以删除该角色并使用 **Viewer** 角色替换。
   - 如果您包含 **Service Account Key Admin** 角色，您可以将其删除。

4.7.8. 通过下载二进制文件安装 OpenShift CLI

您需要安装 CLI（**oc**） 来使用命令行界面与 OpenShift Container Platform 进行交互。您可在 Linux、Windows 或 macOS 上安装 **oc**。

重要

如果安装了旧版本的 **oc**，则无法使用 OpenShift Container Platform 4.6 中的所有命令。下载并安装新版本的 **oc**。

4.7.8.1. 在 Linux 上安装 OpenShift CLI

您可以按照以下流程在 Linux 上安装 OpenShift CLI（**oc**）二进制文件。

流程

1. 进入到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 在 **Version** 下拉菜单中选择相应的版本。
3. 单击 **OpenShift v4.6 Linux 客户端** 条目旁边的 **Download Now**，再保存文件。
4. 解包存档：

```
$ tar xvzf <file>
```

5. 把 **oc** 二进制代码放到 **PATH** 中的目录中。
   执行以下命令可以查看当前的 **PATH** 设置：

```
$ echo $PATH
```

安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
$ oc <command>
```

4.7.8.2. 在 Windows 上安装 OpenShift CLI

您可以按照以下流程在 Windows 上安装 OpenShift CLI（**oc**）二进制代码。

流程

1. 进入到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 在 **Version** 下拉菜单中选择相应的版本。
3. 单击 **OpenShift v4.6 Windows 客户端**条目旁边的 **Download Now**，再保存文件。
4. 使用 ZIP 程序解压存档。
5. 把 **oc** 二进制代码放到 **PATH** 中的目录中。
   要查看您的 **PATH**，请打开命令提示窗口并执行以下命令：

```
C:\> path
```

安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
C:\> oc <command>
```

4.7.8.3. 在 macOS 上安装 OpenShift CLI

您可以按照以下流程在 macOS 上安装 OpenShift CLI（**oc**）二进制代码。

流程

1. 进入到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 在 **Version** 下拉菜单中选择相应的版本。
3. 单击 **OpenShift v4.6 MacOSX 客户端**条目旁边的 **Download Now**，再保存文件。
4. 解包和解压存档。
5. 将 **oc** 二进制文件移到 PATH 的目录中。
   要查看您的 **PATH**，打开一个终端窗口并执行以下命令：

```
$ echo $PATH
```

安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
$ oc <command>
```

4.7.9. 使用 CLI 登录到集群

您可以通过导出集群 **kubeconfig** 文件，以默认系统用户身份登录集群。**kubeconfig** 文件包含关于集群的信息，供 CLI 用于将客户端连接到正确集群和 API 服务器。该文件特只适用于一个特定的集群，在 OpenShift Container Platform 安装过程中创建。

先决条件

- 已部署了 OpenShift Container Platform 集群。
- 已安装 **oc** CLI。

流程

1. 导出 **kubeadmin** 凭证：

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

2. 使用导出的配置，验证能否成功运行 **oc** 命令：

```
$ oc whoami
```

输出示例

```
system:admin
```

其他资源

- 如需有关访问和了解 OpenShift Container Platform Web 控制台的更多信息，请参阅访问 Web 控制台。

4.7.10. OpenShift Container Platform 的 Telemetry 访问

在 OpenShift Container Platform 4.6 中，默认运行的 Telemetry 服务提供有关集群健康状况和成功更新的指标，需要访问互联网。如果您的集群连接到互联网，Telemetry 会自动运行，而且集群会注册到 OpenShift Cluster Manager。

确认 OpenShift Cluster Manager 清单正确后，可以由 Telemetry 自动维护，也可以使用 OpenShift Cluster Manager 手动维护，使用订阅监控来跟踪帐户或多集群级别的 OpenShift Container Platform 订阅。

其他资源

- 有关 Telemetry 服务的更多信息，请参阅关于远程健康监控。

4.7.11. 后续步骤

- 自定义集群。
- 若有需要，您可以选择不使用远程健康报告。

4.8. 在 GCP 上安装私有集群

在 OpenShift Container Platform 版本 4.6 中，您可以在 Google Cloud Platform（GCP）上将私有集群安装到现有的 VPC 中。安装程序会置备所需基础架构的其余部分，您可以进一步定制这些基础架构。要自定义安装，请在安装集群前修改 **install-config.yaml** 文件中的参数。

4.8.1. 先决条件

- 查看有关 OpenShift Container Platform 安装和更新流程的详细信息。
- 配置 GCP 帐户以托管集群。
- 如果使用防火墙，则必须将其配置为允许集群需要访问的站点。
- 如果不允许系统管理身份和访问管理（IAM），集群管理员可以 手动创建和维护 IAM 凭证。手动模式也可以用于云 IAM API 无法访问的环境中。

4.8.2. 私有集群

您可以部署不公开外部端点的私有 OpenShift Container Platform 集群。私有集群只能从内部网络访问，且无法在互联网中看到。

默认情况下，OpenShift Container Platform 被置备为使用可公开访问的 DNS 和端点。私有集群在部署集群时将 DNS、Ingress Controller 和 API 服务器设置为私有。这意味着，集群资源只能从您的内部网络访问，且不能在互联网中看到。

要部署私有集群，您必须使用符合您的要求的现有网络。您的集群资源可能会在网络中的其他集群间共享。

另外，您必须从可访问您置备的云的 API 服务、您置备的网络上的主机以及可以连接到互联网来获取安装介质的机器上部署私有集群。您可以使用符合这些访问要求的机器，并按照您的公司规定进行操作。例如，该机器可以是云网络中的堡垒主机，也可以是可通过 VPN 访问网络的机器。

4.8.2.1. GCP 中的私有群集

要在 Google Cloud Platform (GCP) 上创建私有集群，您必须提供一个现有的私有 VPC 和子网来托管集群。安装程序还必须能够解析集群所需的 DNS 记录。安装程序只为内部流量配置 Ingress Operator 和 API 服务器。

集群仍然需要访问互联网来访问 GCP API。

安装私有集群时不需要或创建以下项目：

- 公共子网
- 支持公共入口的公共网络负载均衡器
- 与集群的 **baseDomain** 匹配的公共 DNS 区域

安装程序会使用您指定的 **baseDomain** 来创建专用 DNS 区域以及集群所需的记录。集群被配置，以便 Operator 不会为集群创建公共记录，且所有集群机器都放置在您指定的私有子网中。

由于无法根据源标签限制对外部负载均衡器的访问，私有集群只使用内部负载均衡器来允许对内部实例的访问。

内部负载均衡器依赖于实例组而不是网络负载均衡器使用的目标池。安装程序为每个区创建实例组，即使该组中没有实例。

- 集群 IP 地址仅为内部地址。
- 一个转发规则管理 Kubernetes API 和机器配置服务器端口。
- 后端服务由每个区实例组以及 bootstrap 实例组（如果存在）组成。
- 防火墙使用一个只基于内部源范围的规则。

4.8.2.1.1. 限制：

因为负载均衡器功能不同，没有运行对机器配置服务器的健康检查（**/healthz**）。两个内部负载均衡器无法共享一个 IP 地址，但两个网络负载均衡器可以共享一个外部 IP 地址。反之，一个实例的健康状况完全由端口 6443 的 **/readyz** 检查决定。

4.8.3. 关于使用自定义 VPC

在 OpenShift Container Platform 4.6 中，您可以在 Google Cloud Platform（GCP）上将集群部署到现有的 VPC 中。如果这样做，需要使用 VPC 中现有的子网以及路由规则。

通过将 OpenShift Container Platform 部署到现有的 GCP VPC 中，您可能会避开新帐户中的限制，或者更容易地利用公司所设置的操作限制。如果您无法获得创建 VPC 所需的基础架构创建权限，则可以使用这个选项。

4.8.3.1. 使用 VPC 的要求

安装程序将不再创建以下组件：

- VPC
- 子网
- 云路由器
- Cloud NAT

- NAT IP 地址

如果您使用自定义 VPC，您必须为安装程序和集群正确配置它及其子网。安装程序不能为集群分配要使用的网络范围，为子网设置路由表，或者设置类似 DHCP 的 VPC 选项，因此您必须在安装集群前配置它们。

您的 VPC 必须满足以下条件：

- VPC 必须位于您将 OpenShift Container Platform 集群部署到的同一 GCP 项目中。
- 要允许 control plane 和计算机器访问互联网，您必须在子网上配置云 NAT 以允许网络出站数据。这些机器没有公共地址。即使不需要访问互联网，您也必须允许到 VPC 网络的网络出站数据，以获取安装程序和镜像。因为不能在共享子网上配置多个云 NAT，所以安装程序无法配置它。

要确保您提供的子网适合您的环境，安装程序会确认以下信息：

- 您指定的所有子网都存在，并属于您指定的 VPC。
- 子网 CIDR 属于机器的 CIDR。
- 您必须提供一个子网来部署集群 control plane 和计算机器。您可以对两种机器类型使用相同的子网。

如果您销毁使用现有 VPC 的集群，VPC 不会被删除。

4.8.3.2. 权限划分

从 OpenShift Container Platform 4.3 开始，您不需要安装程序置备的基础架构集群部署所需的所有权限。这与您所在机构可能已有的权限划分类似：不同的个人可以在您的云中创建不同的资源。例如，您可以创建针对于特定应用程序的对象，如实例、存储桶和负载均衡器，但不能创建与网络相关的组件，如 VPC 、子网或入站规则。

您在创建集群时使用的 GCP 凭证不需要 VPC 和 VPC 中的核心网络组件（如子网、路由表、互联网网关、NAT 和 VPN）所需的网络权限。您仍然需要获取集群中的机器需要的应用程序资源的权限，如负载均衡器 、安全组、存储和节点。

4.8.3.3. 集群间隔离

如果您将 OpenShift Container Platform 部署到现有网络中，则集群服务的隔离由防火墙规则保留，该规则使用集群的基础架构 ID 来引用集群中的机器。仅允许集群中的流量。

如果您将多个集群部署到同一个 VPC 中，则以下组件可能会在集群间共享访问权限：

- API，通过一个外部发布策略全局可用，或通过一个内部发布策略在网络中可用
- 调试工具，如对机器 CIDR 开放的用于 SSH 和 ICMP 访问的 VM 实例上的端口

4.8.4. OpenShift Container Platform 的互联网访问

在 OpenShift Container Platform 4.6 中，您需要访问互联网来安装集群。

您必须具有以下互联网访问权限：

- 访问 OpenShift Cluster Manager 以下载安装程序并执行订阅管理。如果集群可以访问互联网，并且没有禁用 Telemetry，该服务会自动授权您的集群。

- 访问 Quay.io，以获取安装集群所需的软件包。
- 获取执行集群更新所需的软件包。

重要

如果您的集群无法直接访问互联网，则可以在置备的某些类基础架构上执行受限网络安装。在此过程中，您要下载所需的内容，并使用它在镜像 registry（mirror registry） 中填充安装集群并生成安装程序所需的软件包。对于某些安装类型，集群要安装到的环境不需要访问互联网。在更新集群之前，要更新 registry 镜像系统中的内容。

4.8.5. 生成 SSH 私钥并将其添加到代理中

如果要在集群上执行安装调试或灾难恢复，则必须为 **ssh-agent** 和安装程序提供 SSH 密钥。您可以使用此密钥访问公共集群中的 bootstrap 机器来排除安装问题。

注意

在生产环境中，您需要进行灾难恢复和调试。

您可以使用此密钥以 **core** 用户身份通过 SSH 连接到 master 节点。在部署集群时，此密钥会添加到 **core** 用户的 **~/.ssh/authorized_keys** 列表中。

注意

您必须使用一个本地密钥，而不要使用在特定平台上配置的密钥，如 AWS 密钥对。

流程

1. 如果还没有为计算机上免密码身份验证而配置的 SSH 密钥，请创建一个。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ ssh-keygen -t ed25519 -N '' \
  -f <path>/<file_name> 1
```

1 指定新 SSH 密钥的路径和文件名，如 **~/.ssh/id_rsa**。如果您已有密钥对，请确保您的公钥位于 **~/.ssh** 目录中。

运行此命令会在指定的位置生成不需要密码的 SSH 密钥。

注意

如果您计划在 **x86_64** 架构中安装使用 FIPS 验证的/Modules in Process 加密库的 OpenShift Container Platform 集群，不要创建使用 **ed25519** 算法的密钥。反之，创建一个使用 **rsa** 或 **ecdsa** 算法的密钥。

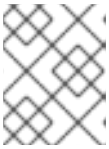
2. 作为后台任务启动 **ssh-agent** 进程：

```
$ eval "$(ssh-agent -s)"
```

输出示例

-

```
Agent pid 31874
```



注意

如果您的集群采用 FIPS 模式，则只使用 FIPS 兼容算法来生成 SSH 密钥。密钥必须是 RSA 或 ECDSA。

3. 将 SSH 私钥添加到 **ssh-agent**：

```
$ ssh-add <path>/<file_name> 1
```

输出示例

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

1 指定 SSH 私钥的路径和文件名，如 **~/.ssh/id_rsa**

4. 将 **GOOGLE_APPLICATION_CREDENTIALS** 环境变量设置为服务帐户私钥文件的完整路径。

```
$ export GOOGLE_APPLICATION_CREDENTIALS="<your_service_account_file>"
```

5. 验证是否应用了凭证。

```
$ gcloud auth list
```

后续步骤

- 在安装 OpenShift Container Platform 时，为安装程序提供 SSH 公钥。

4.8.6. 获取安装程序

在安装 OpenShift Container Platform 之前，将安装文件下载到本地计算机上。

先决条件

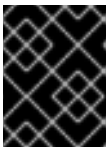
- 运行 Linux 或 macOS 的计算机，本地磁盘空间为 500 MB

流程

1. 访问 OpenShift Cluster Manager 站点的 Infrastructure Provider 页面。如果您有红帽帐号，请使用自己的凭证登录。如果没有，请创建一个帐户。
2. 选择您的基础架构供应商。
3. 进入适用于您的安装类型的页面，下载您的操作系统的安装程序，并将文件放在要保存安装配置文件的目录中。。

重要

安装程序会在用来安装集群的计算机上创建若干文件。在完成集群安装后，您必须保留安装程序和安装程序所创建的文件。这两个文件都需要删除集群。



重要

删除安装程序创建的文件不会删除您的集群，即使集群在安装过程中失败也是如此。要删除集群，为特定云供应商完成 OpenShift Container Platform 卸载流程。

4. 提取安装程序。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ tar xvf openshift-install-linux.tar.gz
```

5. 从 Red Hat OpenShift Cluster Manager 下载安装 pull secret 。通过此 pull secret，您可以进行所含授权机构提供的服务的身份验证，这些服务包括为 OpenShift Container Platform 组件提供容器镜像的 Quay.io。

4.8.7. 手动创建安装配置文件

对于只能从内部网络访问且不能在互联网中看到的私有 OpenShift Container Platform 集群安装，您必须手动生成安装配置文件。

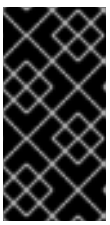
先决条件

- 获取 OpenShift Container Platform 安装程序和集群的访问令牌。

流程

1. 创建用来存储您所需的安装资产的安装目录：

```
$ mkdir <installation_directory>
```



重要

您必须创建目录。一些安装信息，如 bootstrap X.509 证书，有较短的过期间隔，因此不要重复使用安装目录。如果要重复使用另一个集群安装中的个别文件，可以将其复制到您的目录中。但是，一些安装数据的文件名可能会在发行版本之间有所改变。从 OpenShift Container Platform 老版本中复制安装文件时要格外小心。

2. 自定义以下 **install-config.yaml** 文件模板，并将它保存到 **<installation_directory>** 中。

注意

此配置文件必须命名为 **install-config.yaml**。

3. 备份 **install-config.yaml** 文件，以便用于安装多个集群。

重要

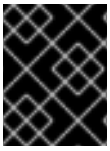
install-config.yaml 文件会在安装过程的下一步骤中消耗掉。现在必须备份它。

4.8.7.1. 安装配置参数

在部署 OpenShift Container Platform 集群前，您可以提供参数值，以描述托管集群的云平台的帐户并选择性地自定义集群平台。在创建 **install-config.yaml** 安装配置文件时，您可以通过命令行来提供所需的参数的值。如果要自定义集群，可以修改 **install-config.yaml** 文件来提供关于平台的更多信息。

注意

安装之后，您无法修改 **install-config.yaml** 文件中的这些参数。



重要

openshift-install 命令不验证参数的字段名称。如果指定了不正确的名称，则不会创建相关的文件或对象，且不会报告错误。确保所有指定的参数的字段名称都正确。

4.8.7.1.1. 所需的配置参数

下表描述了所需的安装配置参数：

表 4.25. 所需的参数

参数	描述	值
apiVersion	**install-config.yaml** 内容的 API 版本。当前版本是 **v1**。安装程序还可能支持旧的 API 版本。	字符串
baseDomain	云供应商的基域。此基础域用于创建到 OpenShift Container Platform 集群组件的路由。集群的完整 DNS 名称是 **baseDomain** 和 **metadata.name** 参数值的组合，其格式为 **<metadata.name>.<baseDomain>**。	完全限定域名或子域名，如 **example.com**。
metadata	Kubernetes 资源 **ObjectMeta**，其中只消耗 **name** 参数。	对象
metadata.name	集群的名称。集群的 DNS 记录是 **{{.metadata.name}}.{{.baseDomain}}** 的子域。	小写字母,连字符(-)和句点(.)的字符串，如 **dev**。
platform	执行安装的具体平台配置：**aws**、**baremetal**、**azure**、**openstack**、**ovirt**、**vsphere**。有关 **platform.<platform>** 参数的额外信息，请参考下表来了解您的具体平台。	对象

参数	描述	值

pullSecret	从 Red Hat OpenShift Cluster Manager 获取 pull secret，验证从 Quay.io 等服务中下载 OpenShift Container Platform 组件的容器镜像。	<pre>{
		  "auths":{
		  "cloud.openshift.com":{
		  "auth":"b3Blb=",
		  "email":"you@example.com"
		  },
		  "quay.io":{
		  "auth":"b3Blb=",
		  "email":"you@example.com"
		  }
		  }
		}</pre>

4.8.7.1.2. 网络配置参数

您可以根据现有网络基础架构的要求自定义安装配置。例如，您可以扩展集群网络的 IP 地址块，或者提供不同于默认值的不同 IP 地址块。

只支持 IPv4 地址。

表 4.26. 网络参数

参数	描述	值
networking	集群网络的配置。	对象 **注意** 您不能在安装后修改 **networking** 对象指定的参数。
networking.networkType	要安装的集群网络供应商 Container Network Interface（CNI）插件。	**OpenShiftSDN** 或 **OVNKubernetes**。默认值为 **OpenShiftSDN**。

networking.clusterNetwork	pod 的 IP 地址块。
	
	默认值为 **10.128.0.0/14**，主机前缀为 **/23**。
	
	如果您指定多个 IP 地址块，则块不得互相重叠。
		一个对象数组。例如：
		
		<pre>networking:
		  clusterNetwork:
		  - cidr: 10.128.0.0/14
		  hostPrefix: 23</pre>

参数	描述	值
networking.clusterNetwork.cidr	使用 **networking.clusterNetwork** 时需要此项。IP 地址块。 一个 IPv4 网络。	使用 CIDR 形式的 IP 地址块。IPv4 块的前缀长度介于 **0** 到 **32** 之间。
networking.clusterNetwork.hostPrefix	分配给每个单独节点的子网前缀长度。例如，如果 **hostPrefix** 设为 **23**，则每个节点从所给的 **cidr** 中分配一个 **/23** 子网。**hostPrefix** 值 **23** 提供 510（2^(32 - 23)- 2）个 pod IP 地址。	子网前缀。 默认值为 **23**。

networking.serviceNetwork	服务的 IP 地址块。默认值为 **172.30.0.0/16**。
	
	OpenShift SDN 和 OVN-Kubernetes 网络供应商只支持服务网络的一个 IP 地址块。
		CIDR 格式具有 IP 地址块的数组。例如：
		
		`networking:`
		`  serviceNetwork:`
		`  - 172.30.0.0/16`

networking.machineNetwork	机器的 IP 地址块。
	
	如果您指定多个 IP 地址块，则块不得互相重叠。
		一个对象数组。例如：
		
		`networking:`
		`  machineNetwork:`
		`  - cidr: 10.0.0.0/16`

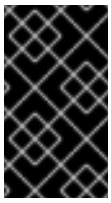
networking.machineNetwork.cidr	使用 **networking.machineNetwork** 时需要。IP 地址块。libvirt 以外的所有平台的默认值为 **10.0.0.0/16**。对于 libvirt，默认值为 **192.168.126.0/24**。	CIDR 表示法中的 IP 网络块。 例如： **10.0.0.0/16**。 **注意** 将 **networking.machineNetwork** 设置为与首选 NIC 所在的 CIDR 匹配。

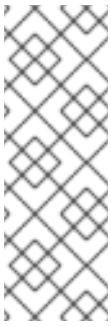
4.8.7.1.3. 可选配置参数

下表描述了可选安装配置参数：

表 4.27. 可选参数

参数	描述	值
additionalTrustBundle	添加到节点可信证书存储中的 PEM 编码 X.509 证书捆绑包。配置了代理时，也可以使用这个信任捆绑包。	字符串

参数	描述	值
compute	组成计算节点的机器的配置。	machine-pool 对象的数组。详情请查看以下"Machine-pool"表。
compute.architecture	决定池中机器的指令集合架构。目前不支持异构集群，因此所有池都必须指定相同的架构。有效值为 **amd64** （默认值）。	字符串
compute.hyperthreading	是否在计算机器上启用或禁用并发多线程或**超线程**。默认情况下，启用并发多线程以提高机器内核的性能。  **重要** 如果禁用并发多线程，请确保在容量规划时考虑到机器性能可能会显著降低的问题。	**Enabled** 或 **Disabled**
compute.name	使用 **compute** 时需要此值。机器池的名称。	**worker**
compute.platform	使用 **compute** 时需要此值。使用此参数指定托管 worker 机器的云供应商。此参数值必须与 **controlPlane.platform** 参数值匹配。	**aws**、**azure**、**gcp**、**openstack**、**ovirt**、**vsphere** 或 **{}**
compute.replicas	要置备的计算机器数量，也称为 worker 机器。	大于或等于 **2** 的正整数。默认值为 **3**。
controlPlane	组成 control plane 的机器的配置。	**MachinePool** 对象的数组。详情请查看以下"Machine-pool"表。
controlPlane.architecture	决定池中机器的指令集合架构。目前不支持异构集群，因此所有池都必须指定相同的架构。有效值为 **amd64** （默认值）。	字符串

参数	描述	值
controlPlane.hyperthreading	是否在 control plane 机器上启用或禁用并发多线程或**超线程**。默认情况下，启用并发多线程以提高机器内核的性能。  **重要** 如果禁用并发多线程，请确保在容量规划时考虑到机器性能可能会显著降低的问题。	**Enabled** 或 **Disabled**
controlPlane.name	使用 **controlPlane** 时需要。机器池的名称。	**master**
controlPlane.platform	使用 **controlPlane** 时需要。使用此参数指定托管 control plane 机器的云供应商。此参数值必须与 **compute.platform** 参数值匹配。	**aws**、**azure**、**gcp**、**openstack**、**ovirt**、**vsphere** 或 **{}**
controlPlane.replicas	要置备的 control plane 机器数量。	唯一支持的值是 **3**，它是默认值。
credentialsMode	Cloud Credential Operator（CCO）模式。如果没有指定任何模式，CCO 会动态地尝试决定提供的凭证的功能，在支持多个模式的平台上使用 mint 模式。  **注意** 不是所有 CCO 模式都支持所有云供应商。如需有关 CCO 模式的更多信息，请参阅*Red Hat Operator 参考指南*内容中的 *Cloud Credential Operator* 条目。	**Mint**、**Passthrough**、**Manual** 或空字符串(**""**)。

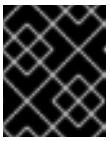
参数	描述	值
fips	启用或禁用 FIPS 模式。默认为 **false**（禁用）。如果启用了 FIPS 模式，运行 OpenShift Container Platform 的 Red Hat Enterprise Linux CoreOS (RHCOS) 机器会绕过默认的 Kubernetes 加密套件，并使用由 RHCOS 提供的加密模块。  **重要** 只有在 **x86_64** 架构中的 OpenShift Container Platform 部署支持 FIPS 验证的/Modules in Process 加密库。  **注意** 如果使用 Azure File 存储，则无法启用 FIPS 模式。	**false** 或 **true**
imageContentSources	release-image 内容的源和仓库。	对象数组。包括一个 **source** 以及可选的 **mirrors**，如下表所示。
imageContentSources.source	使用 **imageContentSources** 时需要。指定用户在镜像拉取规格中引用的仓库。	字符串
imageContentSources.mirrors	指定可能还包含同一镜像的一个或多个仓库。	字符串数组
publish	如何发布或公开集群的面向用户的端点，如 Kubernetes API、OpenShift 路由。	**Internal** 或 **External**。把 **publish** 设置为 **Internal** 以部署一个私有集群，它不能被互联网访问。默认值为 **External**。

sshKey	用于验证集群机器访问的 SSH 密钥或密钥。
	
	 
	注意
	
	对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 **ssh-agent** 进程使用的 SSH 密钥。
		一个或多个密钥。例如：
		
		`sshKey:`
		`  <key1>`
		`  <key2>`
		`  <key3>`

4.8.7.1.4. 其他 Google Cloud Platform（GCP）配置参数

下表中描述了额外的 GCP 配置参数：

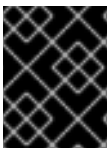
表 4.28. 额外的 GCP 参数

参数	描述	值
platform.gcp.network	要将集群部署到的现有 VPC 的名称。	字符串。
platform.gcp.region	托管集群的 GCP 区域的名称。	任何有效的区域名称，如 **us-central1**。
platform.gcp.type	GCP 机器类型。	GCP 机器类型。
platform.gcp.zones	安装程序在其中为特定 MachinePool 创建机器的可用区。	有效的 GCP 可用区 列表，如 **us-central1-a**，在一个 YAML 序列中。
platform.gcp.controlPlaneSubnet	要将 control plane 机器部署到的 VPC 中现有子网的名称。	子网名称。
platform.gcp.computeSubnet	您要将计算机器部署到的 VPC 中现有子网的名称。	子网名称。
platform.gcp.licenses	必须应用到计算镜像的许可证 URL 列表。  **重要** License **参数** 是一个已弃用的字段，嵌套虚拟化默认为启用。不建议使用此字段。	许可证 API 可获得的任何许可证，如启用 嵌套虚拟化 的许可证。您不能将此参数与生成预构建镜像的机制一起使用。使用许可证 URL 强制安装程序复制源镜像，然后再使用。
platform.gcp.osDisk.diskSizeGB	以 GB (GB) 为单位的磁盘大小。	16 GB 到 65536 GB 之间的任何大小。

参数	描述	值
platform.gcp.osDisk.diskType	磁盘类型。	默认 **pd-ssd** 或 **pd-standard** 磁盘类型。control plane 节点必须是 **pd-ssd** 磁盘类型。worker 节点可以是其中任一个类型。

4.8.7.2. GCP 的自定义 install-config.yaml 文件示例

您可以自定义 **install-config.yaml** 文件，以指定有关 OpenShift Container Platform 集群平台的更多信息，或修改所需参数的值。



重要

此示例 YAML 文件仅供参考。您必须使用安装程序来获取 **install-config.yaml** 文件，并且修改该文件。

```
apiVersion: v1
baseDomain: example.com (1)
controlPlane: (2) (3)
  hyperthreading: Enabled (4)
  name: master
  platform:
    gcp:
      type: n2-standard-4
      zones:
      - us-central1-a
      - us-central1-c
      osDisk:
        diskType: pd-ssd
        diskSizeGB: 1024
  replicas: 3
compute: (5) (6)
- hyperthreading: Enabled (7)
  name: worker
  platform:
    gcp:
      type: n2-standard-4
      zones:
      - us-central1-a
      - us-central1-c
      osDisk:
        diskType: pd-standard
        diskSizeGB: 128
  replicas: 3
metadata:
  name: test-cluster (8)
networking:
  clusterNetwork:
  - cidr: 10.128.0.0/14
    hostPrefix: 23
```

```
  machineNetwork:
  - cidr: 10.0.0.0/16
  networkType: OpenShiftSDN
  serviceNetwork:
  - 172.30.0.0/16
platform:
  gcp:
    projectID: openshift-production 9
    region: us-central1 10
    network: existing_vpc 11
    controlPlaneSubnet: control_plane_subnet 12
    computeSubnet: compute_subnet 13
pullSecret: '{"auths": ...}' 14
fips: false 15
sshKey: ssh-ed25519 AAAA... 16
publish: Internal 17
```

1 8 9 10 14 必需。安装程序会提示您输入这个值。

2 5 如果没有提供这些参数和值，安装程序会提供默认值。

3 6 **controlPlane** 部分是一个单映射，但 **compute** 部分是一系列映射。为满足不同数据结构的要求，**compute** 部分的第一行必须以连字符 **-** 开头，**controlPlane** 部分的第一行则不可以连字符开头。只使用一个 control plane 池。

4 7 是否要启用或禁用并发多线程或**超线程**。默认情况下，启用并发多线程以提高机器内核的性能。您可以通过将参数值设为 **Disabled** 来禁用。如果您在某些集群机器上禁用并发多线程，则必须在所有集群机器上禁用。

重要

如果禁用并发多线程，请确保在容量规划时考虑到机器性能可能会显著降低的问题。如果您禁用并发多线程，请为您的机器使用较大的类型，如 **n1-standard-8**。

11 指定现有 VPC 的名称。

12 指定要将 control plane 机器部署到的现有子网的名称。该子网必须属于您指定的 VPC。

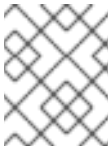
13 指定要将计算机器部署到的现有子网的名称。该子网必须属于您指定的 VPC。

15 是否启用或禁用 FIPS 模式。默认情况下不启用 FIPS 模式。如果启用了 FIPS 模式，运行 OpenShift Container Platform 的 Red Hat Enterprise Linux CoreOS (RHCOS) 机器会绕过默认的 Kubernetes 加密套件，并使用由 RHCOS 提供的加密模块。

重要

只有在 **x86_64** 架构中的 OpenShift Container Platform 部署支持 FIPS 验证的/Modules in Process 加密库。

16 您可以选择提供您用来访问集群中机器的 **sshKey** 值。



注意

对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 **ssh-agent** 进程使用的 SSH 密钥。

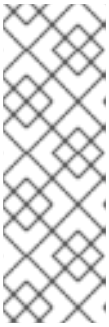
17 如何发布集群的面向用户的端点。把 **publish** 设置为 **Internal** 以部署一个私有集群，它不能被互联网访问。默认值为 **External**。

4.8.7.3. 在安装过程中配置集群范围代理

生产环境可能会拒绝直接访问互联网，而是提供 HTTP 或 HTTPS 代理。您可以通过在 **install-config.yaml** 文件中配置代理设置，将新的 OpenShift Container Platform 集群配置为使用代理。

先决条件

- 您有一个现有的 **install-config.yaml** 文件。
- 您检查了集群需要访问的站点，并决定是否需要绕过代理。默认情况下代理所有集群出口流量，包括对托管云供应商 API 的调用。您需要将站点添加到 **Proxy** 对象的 **spec.noProxy** 字段来绕过代理。



注意

Proxy 对象 **status.noProxy** 字段使用安装配置中的 **networking.machineNetwork[].cidr**、**networking.clusterNetwork[].cidr** 和 **networking.serviceNetwork[]** 字段的值填充。

对于在 Amazon Web Services(AWS)、Google Cloud Platform(GCP)、Microsoft Azure 和 Red Hat OpenStack Platform(RHOSP)上安装, **Proxy** 对象 **status.noProxy** 字段也会使用实例元数据端点填充(**169.254.169.254**)。

流程

1. 编辑 **install-config.yaml** 文件并添加代理设置。例如：

```
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: example.com 3
additionalTrustBundle: | 4
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
...
```

1 用于创建集群外 HTTP 连接的代理 URL。URL 必须是 **http**。

2 用于创建集群外 HTTPS 连接的代理 URL。

3 要排除在代理中的目标域名、IP 地址或其他网络 CIDR 的逗号分隔列表。在域前面加 **.** 来仅匹配子域。例如： **.y.com** 匹配 **x.y.com**，但不匹配 **y.com**。使用 ***** 绕过所有目的地的代理。

4 如果提供，安装程序会在 **openshift-config 命名空间中生成名为 user-ca- bundle** 的配置映射来保存额外的 CA 证书。如果您提供 **additionalTrustBundle** 和至少一个代理设置，则 **Proxy** 对象会被配置为引用 **trustedCA** 字段中的 **user-ca-bundle** 配置映射。然后，Cluster Network Operator 会创建一个 **trusted-ca-bundle** 配置映射，该配置映射将为 **trustedCA** 参数指定的内容与 RHCOS 信任捆绑包合并。**additionalTrustBundle** 字段是必需的，除非代理的身份证书由来自 RHCOS 信任捆绑包的颁发机构签名。

注意

安装程序不支持代理的 **readinessEndpoints** 字段。

2. 保存该文件，并在安装 OpenShift Container Platform 时引用。

安装程序会创建一个名为 **cluster** 的集群范围代理，该代理使用提供的 **install-config.yaml** 文件中的代理设置。如果没有提供代理设置，仍然会创建一个 **cluster Proxy** 对象，但它会有一个空 **spec**。

注意

只支持名为 **cluster** 的 **Proxy** 对象，且无法创建额外的代理。

4.8.8. 部署集群

您可以在兼容云平台中安装 OpenShift Container Platform。

重要

安装程序的 **create cluster** 命令只能在初始安装过程中运行一次。

先决条件

- 配置托管集群的云平台的帐户。
- 获取 OpenShift Container Platform 安装程序以及集群的 pull secret。

流程

1. 更改为包含安装程序的目录并初始化集群部署：

```
$ ./openshift-install create cluster --dir <installation_directory> \ 1
    --log-level=info 2
```

1 对于 **<installation_directory>**，请指定

2 要查看不同的安装详情，请指定 **warn**、**debug** 或 **error**，而不要指定 **info**。

注意

如果您在主机上配置的云供应商帐户没有足够的权限来部署集群，安装过程将会停止，并且显示缺少的权限。

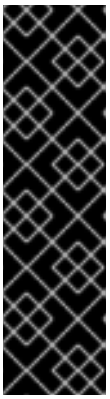
集群部署完成后，终端会显示访问集群的信息，包括指向其 Web 控制台的链接和 **kubeadmin** 用户的凭证。

输出示例

```
...
INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run 'export
KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'
INFO Access the OpenShift web-console here: https://console-openshift-
console.apps.mycluster.example.com
INFO Login to the console with user: "kubeadmin", and password: "4vYBz-Ee6gm-ymBZj-
Wt5AL"
INFO Time elapsed: 36m22s
```

注意

当安装成功时，集群访问和凭证信息还会输出到 **<installation_directory>/.openshift_install.log**。



重要

- 安装程序生成的 Ignition 配置文件包含在 24 小时后过期的证书，然后在过期时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外情况是，您需要手动批准待处理的 **node-bootstrapper** 证书签名请求（CSR）来恢复 kubelet 证书。如需更多信息，请参阅*从过期的 control plane 证书中恢复*的文档。
- 建议您在生成 12 小时后使用 Ignition 配置文件，因为集群安装后 24 小时证书从 16 小时轮转至 22 小时。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中运行证书更新时避免安装失败。

重要

您不得删除安装程序或安装程序所创建的文件。需要这两者才能删除集群。

4.8.9. 通过下载二进制文件安装 OpenShift CLI

您需要安装 CLI（**oc**） 来使用命令行界面与 OpenShift Container Platform 进行交互。您可在 Linux、Windows 或 macOS 上安装 **oc**。

重要

如果安装了旧版本的 **oc**，则无法使用 OpenShift Container Platform 4.6 中的所有命令。下载并安装新版本的 **oc**。

4.8.9.1. 在 Linux 上安装 OpenShift CLI

您可以按照以下流程在 Linux 上安装 OpenShift CLI（**oc**）二进制文件。

流程

1. 进入到红帽客户门户网站上的 OpenShift Container Platform 下载页面。

2. 在 **Version** 下拉菜单中选择相应的版本。
3. 单击 **OpenShift v4.6 Linux 客户端** 条目旁边的 **Download Now**，再保存文件。
4. 解包存档：

```
$ tar xvzf <file>
```

5. 把 **oc** 二进制代码放到 **PATH** 中的目录中。
   执行以下命令可以查看当前的 **PATH** 设置：

```
$ echo $PATH
```

安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
$ oc <command>
```

4.8.9.2. 在 Windows 上安装 OpenShift CLI

您可以按照以下流程在 Windows 上安装 OpenShift CLI（**oc**）二进制代码。

流程

1. 进入到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 在 **Version** 下拉菜单中选择相应的版本。
3. 单击 **OpenShift v4.6 Windows 客户端**条目旁边的 **Download Now**，再保存文件。
4. 使用 ZIP 程序解压存档。
5. 把 **oc** 二进制代码放到 **PATH** 中的目录中。
   要查看您的 **PATH**，请打开命令提示窗口并执行以下命令：

```
C:\> path
```

安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
C:\> oc <command>
```

4.8.9.3. 在 macOS 上安装 OpenShift CLI

您可以按照以下流程在 macOS 上安装 OpenShift CLI（**oc**）二进制代码。

流程

1. 进入到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 在 **Version** 下拉菜单中选择相应的版本。
3. 单击 **OpenShift v4.6 MacOSX 客户端**条目旁边的 **Download Now**，再保存文件。
4. 解包和解压存档。

5. 将 **oc** 二进制文件移到 PATH 的目录中。
   要查看您的 **PATH**，打开一个终端窗口并执行以下命令：

```
$ echo $PATH
```

安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
$ oc <command>
```

4.8.10. 使用 CLI 登录到集群

您可以通过导出集群 **kubeconfig** 文件，以默认系统用户身份登录集群。**kubeconfig** 文件包含关于集群的信息，供 CLI 用于将客户端连接到正确集群和 API 服务器。该文件特只适用于一个特定的集群，在 OpenShift Container Platform 安装过程中创建。

先决条件

- 已部署了 OpenShift Container Platform 集群。
- 已安装 **oc** CLI。

流程

1. 导出 **kubeadmin** 凭证：

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

   1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

2. 使用导出的配置，验证能否成功运行 **oc** 命令：

```
$ oc whoami
```

输出示例

```
system:admin
```

其他资源

- 如需有关访问和了解 OpenShift Container Platform Web 控制台的更多信息，请参阅访问 Web 控制台。

4.8.11. OpenShift Container Platform 的 Telemetry 访问

在 OpenShift Container Platform 4.6 中，默认运行的 Telemetry 服务提供有关集群健康状况和成功更新的指标，需要访问互联网。如果您的集群连接到互联网，Telemetry 会自动运行，而且集群会注册到 OpenShift Cluster Manager。

确认 OpenShift Cluster Manager 清单正确后，可以由 Telemetry 自动维护，也可以使用 OpenShift Cluster Manager 手动维护，使用订阅监控来跟踪帐户或多集群级别的 OpenShift Container Platform 订阅。

其他资源

- 有关 Telemetry 服务的更多信息，请参阅关于远程健康监控。

4.8.12. 后续步骤

- 自定义集群。
- 若有需要，您可以选择不使用远程健康报告。

4.9. 使用 DEPLOYMENT MANAGER 模板在 GCP 中的用户置备的基础架构上安装集群

在 OpenShift Container Platform 版本 4.6 中，您可以使用您提供的基础架构在 Google Cloud Platform（GCP）上安装集群。

此处概述了进行用户提供基础架构安装的步骤。提供的几个 Deployment Manager 模板可协助完成这些步骤，也可帮助您自行建模。您还可以自由选择通过其他方法创建所需的资源。

重要

进行用户置备的基础架构安装的步骤仅作为示例。使用您提供的基础架构安装集群需要了解云供应商和 OpenShift Container Platform 安装过程。提供的几个 Deployment Manager 模板可帮助完成这些步骤，或者帮助您自行建模。您也可以自由选择通过其他方法创建所需的资源；模板仅作示例之用。

4.9.1. 先决条件

- 查看有关 OpenShift Container Platform 安装和更新 流程的详细信息。
- 如果使用防火墙并计划使用遥测（telemetry），您必须将防火墙配置为允许集群需要访问的站点。
- 如果不允许系统管理身份和访问管理（IAM），集群管理员可以 手动创建和维护 IAM 凭证。手动模式也可以用于云 IAM API 无法访问的环境中。

注意

如果您要配置代理，请务必也要查看此站点列表。

4.9.2. 证书签名请求管理

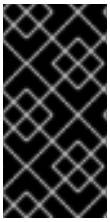
在使用您置备的基础架构时，集群只能有限地访问自动机器管理，因此您必须提供一种在安装后批准集群证书签名请求 (CSR) 的机制。**kube-controller-manager** 只能批准 kubelet 客户端 CSR。 **machine-approver** 无法保证使用 kubelet 凭证请求的提供证书的有效性，因为它不能确认是正确的机器发出了该请求。您必须决定并实施一种方法，以验证 kubelet 提供证书请求的有效性并进行批准。

4.9.3. OpenShift Container Platform 的互联网访问

在 OpenShift Container Platform 4.6 中，您需要访问互联网来安装集群。

您必须具有以下互联网访问权限：

- 访问 OpenShift Cluster Manager 以下载安装程序并执行订阅管理。如果集群可以访问互联网，并且没有禁用 Telemetry，该服务会自动授权您的集群。
- 访问 Quay.io，以获取安装集群所需的软件包。
- 获取执行集群更新所需的软件包。



重要

如果您的集群无法直接访问互联网，则可以在置备的某些类基础架构上执行受限网络安装。在此过程中，您要下载所需的内容，并使用它在镜像 registry（mirror registry） 中填充安装集群并生成安装程序所需的软件包。对于某些安装类型，集群要安装到的环境不需要访问互联网。在更新集群之前，要更新 registry 镜像系统中的内容。

4.9.4. 配置 GCP 项目

在安装 OpenShift Container Platform 之前，您必须配置 Google Cloud Platform (GCP) 项目来托管它。

4.9.4.1. 创建一个 GCP 项目

为了安装 OpenShift Container Platform，您必须在您的 Google Cloud Platform (GCP) 账户中创建一个项目来托管它。

流程

- 创建一个项目来托管您的 OpenShift Container Platform 集群。请参阅 GCP 文档中的创建和管理项目。

重要

如果您使用安装程序置备的基础架构，您的 GCP 项目必须使用 Premium Network Service Tier。使用安装程序安装的集群不支持 Standard Network Service Tier。安装程序为 **api-int.<cluster_name>.<base_domain>** URL 配置内部负载均衡；内部负载均衡需要 Premium Tier。

4.9.4.2. 在 GCP 中启用 API 服务

Google Cloud Platform (GCP) 项目需要访问一些 API 服务来完成 OpenShift Container Platform 安装。

先决条件

- 创建了用于托管集群的项目。

流程

- 在托管集群的项目中启用如下所需的 API 服务。请参阅 GCP 文档中的启用服务。

表 4.29. 所需的 API 服务

API 服务	控制台服务名称
Cloud Deployment Manager V2 API	**deploymentmanager.googleapis.com**

API 服务	控制台服务名称
compute Engine API	**compute.googleapis.com**
Google Cloud API	**cloudapis.googleapis.com**
Cloud Resource Manager API	**cloudresourcemanager.googleapis.com**
Google DNS API	**dns.googleapis.com**
IAM Service Account Credentials API	**iamcredentials.googleapis.com**
Identity and Access Management (IAM) API	**iam.googleapis.com**
Service Management API	**servicemanagement.googleapis.com**
Service Usage API	**serviceusage.googleapis.com**
Google Cloud Storage JSON API	**storage-api.googleapis.com**
Cloud Storage	**storage-component.googleapis.com**

4.9.4.3. 为 GCP 配置 DNS

要安装 OpenShift Container Platform，您使用的 Google Cloud Platform (GCP) 帐户必须在托管 OpenShift Container Platform 集群的同一项目中有一个专用的公共托管区。此区域必须对域具有权威。DNS 服务为集群外部连接提供集群 DNS 解析和名称查询。

流程

1. 标识您的域或子域，以及注册商（registrar）。您可以转移现有的域和注册商，或通过 GCP 或其他来源获取新的域和注册商。

注意

如果您购买新域，则需要时间来传播相关的 DNS 更改。有关通过 Google 购买域的更多信息，请参阅 Google Domains。

2. 在 GCP 项目中为您的域或子域创建一个公共托管区。请参阅 GCP 文档中的创建公共区。使用合适的根域（如 **openshiftcorp.com**）或子域（如 **clusters.openshiftcorp.com**）。

3. 从托管区记录中提取新的权威名称服务器。请参阅 GCP 文档中的查找您的云 DNS 名称服务器。您通常有四个名称服务器。

4. 更新域所用名称服务器的注册商记录。例如，如果您将域注册到了 Google Domains，请参阅 Google Domains 帮助中的以下主题：如何切换到自定义名称服务器。

5. 如果您将根域迁移到 Google Cloud DNS，请迁移您的 DNS 记录。请参阅 GCP 文档中的 Migrating to Cloud DNS 。

6. 如果您使用子域，请按照您公司的流程将其委派记录添加到父域。这个过程可能包括对您公司的 IT 部门或控制您公司的根域和 DNS 服务的部门提出请求。

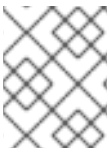
4.9.4.4. GCP 帐户限值

OpenShift Container Platform 集群使用诸多 Google Cloud Platform (GCP) 组件，默认的配额不会影响您安装默认 OpenShift Container Platform 集群的能力。

默认集群中包含三台计算机器和三台 control plane 机器，使用了以下资源。请注意，一些资源只在 bootstrap 过程中需要，会在集群部署后删除。

表 4.30. 默认集群中使用的 GCP 资源

服务	组件	位置	所需的资源总数	bootstrap 后删除的资源
服务帐户	IAM	全局	5	0
防火墙规则	网络	全局	11	1
转发规则	Compute	全局	2	0
健康检查	Compute	全局	2	0
镜像	Compute	全局	1	0
网络	网络	全局	1	0
路由器	网络	全局	1	0
Routes	网络	全局	2	0
子网	Compute	全局	2	0
目标池	网络	全局	2	0



注意

如果在安装过程中存在任何配额不足的情况，安装程序会显示一个错误信息，包括超过哪个配额，以及相关的区域。

请考虑您的集群的实际大小、预定的集群增长以及来自与您的帐户关联的其它集群的使用情况。CPU、静态 IP 地址和持久性磁盘 SSD (storage) 配额是最可能不足的。

如果您计划在以下区域之一部署集群，您将超过最大存储配额，并可能会超过 CPU 配额限制：

- **asia-east2**
- **asia-northeast2**
- **asia-south1**

- **domain-southeast1**
- **europe-north1**
- **europe-west2**
- **europe-west3**
- **europe-west6**
- **northamerica-northeast1**
- **southamerica-east1**
- **us-west2**

您可以从 GCP 控制台增加资源配额，但可能需要提交一个支持问题单。务必提前规划集群大小，以便在安装 OpenShift Container Platform 集群前有足够的时间来等待支持问题单被处理。

4.9.4.5. 在 GCP 中创建服务帐户

OpenShift Container Platform 需要一个 Google Cloud Platform (GCP) 服务帐户，用于访问 Google API 中数据的验证和授权。如果您没有包含项目中所需角色的现有 IAM 服务帐户，您必须创建一个。

先决条件

- 创建了用于托管集群的项目。

流程

1. 在用于托管 OpenShift Container Platform 集群的项目中，创建一个新服务帐户。请参阅 GCP 文档中的创建服务账户。
2. 为服务帐户授予适当的权限。您可以逐一授予权限，也可以为其分配 **Owner** 角色。请参阅将角色授予特定资源的服务帐户。

注意

获得所需权限最简单的方法是把服务帐户设置为项目的拥有者（owner），这意味着该服务帐户对项目有完全的控制权。您必须考虑这样设定所带来的的风险是否可以接受。

3. 以 JSON 格式创建服务帐户密钥。请参阅 GCP 文档中的创建服务帐户密钥。
创建集群时需要该服务帐户密钥。

4.9.4.5.1. 所需的 GCP 权限

如果将 **Owner** 角色附加到您创建的服务帐户，您向该服务帐户授予所有的权限，包括安装 OpenShift Container Platform 所需的权限。要部署 OpenShift Container Platform 集群，服务帐户需要以下权限：
如果您将集群部署到现有的 VPC 中，则服务帐户不需要某些网络权限，如下表所示：

安装程序所需的角色

- Compute Admin

- Security Admin
- Service Account Admin
- Service Account User
- Storage Admin

安装过程中创建网络资源所需的角色

- DNS Administrator

用户置备的 GCP 基础架构所需的角色

- Deployment Manager Editor
- Service Account Key Admin

可选角色

若要使集群为其 Operator 创建新的有限凭证，请添加以下角色：

- Service Account Key Admin

以下角色将应用到 control plane 或计算机器使用的服务帐户：

表 4.31. GCP 服务帐户权限

帐户	角色
Control Plane	**roles/compute.instanceAdmin**
	roles/compute.networkAdmin
	roles/compute.securityAdmin
	roles/storage.admin
	roles/iam.serviceAccountUser
Compute	**roles/compute.viewer**
	roles/storage.admin

4.9.4.6. 支持的 GCP 区域

您可以将 OpenShift Container Platform 集群部署到下列 Google Cloud Platform (GCP) 区域：

- **asia-east1** (Changhua County, Taiwan)
- **asia-east2** (Hong Kong)
- **asia-northeast1** (Tokyo, Japan)

- **asia-northeast2** (Osaka, Japan)
- **asia-northeast3** (Seoul, South Korea)
- **asia-south1** (Mumbai, India)
- **asia-southeast1** (Jurong West, Singapore)
- **asia-southeast2** (Jakarta, India)
- **australia-southeast1** (Sydney, Australia)
- **europe-north1** (Hamina, Finland)
- **europe-west1** (St. Ghislain, Belgium)
- **europe-west2** (London, England, UK)
- **europe-west3** (Frankfurt, Germany)
- **europe-west4** (Eemshaven, Netherlands)
- **europe-west6** (Zürich, Switzerland)
- **northamerica-northeast1** (Montréal, Québec, Canada)
- **southamerica-east1** (São Paulo, Brazil)
- **us-central1** (Council Bluffs, Iowa, USA)
- **us-east1** (Moncks Corner, South Carolina, USA)
- **us-east4** (Ashburn, Northern Virginia, USA)
- **us-west1** (The Dalles, Oregon, USA)
- **us-west2** (Los Angeles, California, USA)
- **us-west3** (Salt Lake City, Utah, USA)
- **us-west4** (Las Vegas, Nevada, USA)

4.9.4.7. 为 GCP 安装和配置 CLI 工具

要使用用户置备的基础架构在 Google Cloud Platform (GCP) 上安装 OpenShift Container Platform，您必须为 GCP 安装和配置 CLI 工具。

先决条件

- 创建了用于托管集群的项目。
- 您创建了服务帐户，并授予其所需的权限。

流程

1. 在 **$PATH** 中安装以下二进制文件：

- **gcloud**
- **gsutil**

请参阅 GCP 文档中的安装最新 Cloud SDK 版本。

2. 使用 **gcloud** 工具及您配置的服务帐户进行身份验证。
请参阅 GCP 文档中的使用服务帐户授权。

4.9.5. 为 GCP 创建安装文件

要使用用户置备的基础架构在 Google Cloud Platform (GCP) 上安装 OpenShift Container Platform，您必须生成并修改安装程序部署集群所需的文件，以便集群只创建要使用的机器。您要生成并自定义 **install-config.yaml** 文件、Kubernetes 清单和 Ignition 配置文件。您也可以选择在安装准备阶段首先设置独立的 **var** 分区。

4.9.5.1. 可选：创建独立 /var 分区

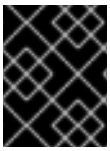
建议安装程序将 OpenShift Container Platform 的磁盘分区保留给安装程序。然而，在有些情况下您可能需要在文件系统的一部分中创建独立分区。

OpenShift Container Platform 支持添加单个分区来将存储附加到 **/var** 分区或 **/var** 的子目录。例如：

- **/var/lib/containers**：保存镜像相关的内容，随着更多镜像和容器添加到系统中，它所占用的存储会增加。
- **/var/lib/etcd**：保存您可能希望保持独立的数据，比如 etcd 存储的性能优化。
- **/var**：保存您希望独立保留的数据，用于特定目的（如审计）。

单独存储 **/var** 目录的内容可方便地根据需要对区域扩展存储，并可以在以后重新安装 OpenShift Container Platform 时保持该数据地完整。使用这个方法，您不必再次拉取所有容器，在更新系统时也无法复制大量日志文件。

因为 **/var** 在进行一个全新的 Red Hat Enterprise Linux CoreOS（RHCOS）安装前必需存在，所以这个流程会在 OpenShift Container Platform 安装过程的 **openshift-install** 准备阶段插入的机器配置来设置独立的 **/var** 分区。



重要

如果按照以下步骤在此流程中创建独立 **/var** 分区，则不需要再次创建 Kubernetes 清单和 Ignition 配置文件，如本节所述。

流程

1. 创建存放 OpenShift Container Platform 安装文件的目录：

```
$ mkdir $HOME/clusterconfig
```

2. 运行 **openshift-install** 在 **manifest** 和 **openshift** 子目录中创建一组文件。在出现提示时回答系统问题：

```
$ openshift-install create manifests --dir $HOME/clusterconfig
```

输出示例

```
? SSH Public Key ...
INFO Credentials loaded from the "myprofile" profile in file "/home/myuser/.aws/credentials"
INFO Consuming Install Config from target directory
INFO Manifests created in: $HOME/clusterconfig/manifests and
$HOME/clusterconfig/openshift
```

3. 可选：确认安装程序在 **clusterconfig/openshift** 目录中创建了清单：

```
$ ls $HOME/clusterconfig/openshift/
```

输出示例

```
99_kubeadmin-password-secret.yaml
99_openshift-cluster-api_master-machines-0.yaml
99_openshift-cluster-api_master-machines-1.yaml
99_openshift-cluster-api_master-machines-2.yaml
...
```

4. 创建 **MachineConfig** 对象并将其添加到 **openshift** 目录中的一个文件中。例如，把文件命名为 **98-var-partition.yaml**，将磁盘设备名称改为 **worker** 系统中存储设备的名称，并根据情况设置存储大小。这个示例将 **/var** 目录放在一个单独的分区中：

```
apiVersion: machineconfiguration.openshift.io/v1
kind: MachineConfig
metadata:
  labels:
    machineconfiguration.openshift.io/role: worker
  name: 98-var-partition
spec:
  config:
    ignition:
      version: 3.1.0
    storage:
      disks:
      - device: /dev/<device_name> 1
        partitions:
        - label: var
          startMiB: <partition_start_offset> 2
          sizeMiB: <partition_size> 3
      filesystems:
        - device: /dev/disk/by-partlabel/var
          path: /var
          format: xfs
    systemd:
      units:
        - name: var.mount 4
          enabled: true
          contents: |
            [Unit]
            Before=local-fs.target
            [Mount]
            What=/dev/disk/by-partlabel/var
```

```
Where=/var
Options=defaults,prjquota 5
[Install]
WantedBy=local-fs.target
```

1. 要分区的磁盘的存储设备名称。
2. 当在引导磁盘中添加数据分区时，推荐最少使用 25000 MiB（Mebibytes）。root 文件系统会自动重新定义大小使其占据所有可用空间（最多到指定的偏移值）。如果没有指定值，或者指定的值小于推荐的最小值，则生成的 root 文件系统会太小，而在以后进行的 RHCOS 重新安装可能会覆盖数据分区的开始部分。
3. 数据分区的大小（以兆字节为单位）。
4. 挂载单元的名称必须与 **Where=** 指令中指定的目录匹配。例如，对于挂载在 **/var/lib/containers** 上的文件系统，该单元必须命名为 **var-lib-containers.mount**。
5. 对于用于容器存储的文件系统，必须启用 **prjquota** 挂载选项。

注意

在创建独立 **/var** 分区时，如果不同的实例类型没有相同的设备名称，则无法将不同的实例类型用于 worker 节点。

5. 再次运行 **openshift-install**，从 **manifest** 和 **openshift** 子目录中的一组文件创建 Ignition 配置：

```
$ openshift-install create ignition-configs --dir $HOME/clusterconfig
$ ls $HOME/clusterconfig/
auth  bootstrap.ign  master.ign  metadata.json  worker.ign
```

现在，您可以使用 Ignition 配置文件作为安装程序的输入来安装 Red Hat Enterprise Linux CoreOS（RHCOS）系统。

4.9.5.2. 创建安装配置文件

您可以自定义在 Google Cloud Platform (GCP) 上安装的 OpenShift Container Platform 集群。

先决条件

- 获取 OpenShift Container Platform 安装程序以及集群的 pull secret。

流程

1. 创建 **install-config.yaml** 文件。
   a. 更改到包含安装程序的目录，再运行以下命令：

```
$ ./openshift-install create install-config --dir <installation_directory> 1
```

1. 对于 **<installation_directory>**，请指定用于保存安装程序所创建的文件的目录名称。

重要

指定一个空目录。一些安装信息，如 bootstrap X.509 证书，有较短的过期间隔，因此不要重复使用安装目录。如果要重复使用另一个集群安装中的个别文件，可以将其复制到您的目录中。但是，一些安装数据的文件名可能会在发行版本之间有所改变。从 OpenShift Container Platform 老版本中复制安装文件时要格外小心。

b. 在提示符处，提供您的云的配置详情：

i. 可选：选择用来访问集群机器的 SSH 密钥。

注意

对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 **ssh-agent** 进程使用的 SSH 密钥。

ii. 选择 **gcp** 作为目标平台。

iii. 如果您没有为计算机上的 GCP 帐户配置服务帐户密钥，则必须从 GCP 获取，并粘贴文件的内容或输入文件的绝对路径。

iv. 选择要在其中置备集群的项目 ID。默认值由您配置的服务帐户指定。

v. 选择要在其中部署集群的区域。

vi. 选择集群要部署到的基域。基域与您为集群创建的公共 DNS 区对应。

vii. 为集群输入一个描述性名称。

viii. 粘贴 Red Hat OpenShift Cluster Manager 中的 pull secret 。

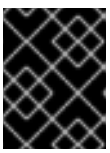
c. 可选：如果您不希望集群置备计算机器，请通过编辑 **install-config.yaml** 文件将 **compute** 池的 **replicas** 设置为 **0** 来清空计算池。

```
compute:
- hyperthreading: Enabled
  name: worker
  platform: {}
  replicas: 0 1
```

1 设置为 **0**。

2. 修改 **install-config.yaml** 文件。您可以在**安装配置参数**部分中找到有关可用参数的更多信息。

3. 备份 **install-config.yaml** 文件，以便用于安装多个集群。



重要

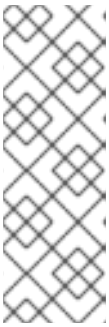
install-config.yaml 文件会在安装过程中消耗掉。如果要重复使用此文件，必须现在备份。

4.9.5.3. 在安装过程中配置集群范围代理

生产环境可能会拒绝直接访问互联网，而是提供 HTTP 或 HTTPS 代理。您可以通过在 **install-config.yaml** 文件中配置代理设置，将新的 OpenShift Container Platform 集群配置为使用代理。

先决条件

- 您有一个现有的 **install-config.yaml** 文件。
- 您检查了集群需要访问的站点，并决定是否需要绕过代理。默认情况下代理所有集群出口流量，包括对托管云供应商 API 的调用。您需要将站点添加到 **Proxy** 对象的 **spec.noProxy** 字段来绕过代理。



注意

Proxy 对象 **status.noProxy** 字段使用安装配置中的 **networking.machineNetwork[].cidr**、**networking.clusterNetwork[].cidr** 和 **networking.serviceNetwork[]** 字段的值填充。

对于在 Amazon Web Services(AWS)、Google Cloud Platform(GCP)、Microsoft Azure 和 Red Hat OpenStack Platform(RHOSP)上安装, **Proxy** 对象 **status.noProxy** 字段也会使用实例元数据端点填充(**169.254.169.254**)。

流程

1. 编辑 **install-config.yaml** 文件并添加代理设置。例如：

```
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: example.com 3
additionalTrustBundle: | 4
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
...
```

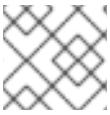
1. 用于创建集群外 HTTP 连接的代理 URL。URL 必须是 **http**。
2. 用于创建集群外 HTTPS 连接的代理 URL。
3. 要排除在代理中的目标域名、IP 地址或其他网络 CIDR 的逗号分隔列表。在域前面加 **.** 来仅匹配子域。例如： **.y.com** 匹配 **x.y.com**，但不匹配 **y.com**。使用 * 绕过所有目的地的代理。
4. 如果提供，安装程序会在 **openshift-config 命名空间中生成名为 user-ca- bundle** 的配置映射来保存额外的 CA 证书。如果您提供 **additionalTrustBundle** 和至少一个代理设置，则 **Proxy** 对象会被配置为引用 **trustedCA** 字段中的 **user-ca-bundle** 配置映射。然后，Cluster Network Operator 会创建一个 **trusted-ca-bundle** 配置映射，该配置映射将为 **trustedCA** 参数指定的内容与 RHCOS 信任捆绑包合并。 **additionalTrustBundle** 字段是必需的，除非代理的身份证书由来自 RHCOS 信任捆绑包的颁发机构签名。

注意

安装程序不支持代理的 **readinessEndpoints** 字段。

2. 保存该文件，并在安装 OpenShift Container Platform 时引用。

安装程序会创建一个名为 **cluster** 的集群范围代理，该代理使用提供的 **install-config.yaml** 文件中的代理设置。如果没有提供代理设置，仍然会创建一个 **cluster Proxy** 对象，但它会有一个空 **spec**。



注意

只支持名为 **cluster** 的 **Proxy** 对象，且无法创建额外的代理。

4.9.5.4. 创建 Kubernetes 清单和 Ignition 配置文件

由于您必须修改一些集群定义文件并要手动启动集群机器，因此您必须生成 Kubernetes 清单和 Ignition 配置文件，集群需要这两项来创建其机器。

安装配置文件转换为 Kubernetes 清单。清单嵌套到 Ignition 配置文件中，稍后用于创建集群。

重要

- 安装程序生成的 Ignition 配置文件包含在 24 小时后过期的证书，然后在过期时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外情况是，您需要手动批准待处理的 **node-bootstrapper** 证书签名请求（CSR）来恢复 kubelet 证书。如需更多信息，请参阅*从过期的 control plane 证书中恢复*的文档。
- 建议您在生成 12 小时后使用 Ignition 配置文件，因为集群安装后 24 小时证书从 16 小时轮转至 22 小时。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中运行证书更新时避免安装失败。

先决条件

- 已获得 OpenShift Container Platform 安装程序。
- 已创建 **install-config.yaml** 安装配置文件。

流程

1. 切换到包含安装程序的目录，并为集群生成 Kubernetes 清单：

```
$ ./openshift-install create manifests --dir <installation_directory> 1
```

1 对于 **<installation_directory>**，请指定含有您创建的 **install-config.yaml** 文件的安装目录。

2. 删除定义 control plane 机器的 Kubernetes 清单文件：

```
$ rm -f <installation_directory>/openshift/99_openshift-cluster-api_master-machines-*.yaml
```

通过删除这些文件，您可以防止集群自动生成 control plane 机器。

3. 可选：如果您不希望集群置备计算机器，请删除定义 worker 机器的 Kubernetes 清单文件：

```
$ rm -f <installation_directory>/openshift/99_openshift-cluster-api_worker-machineset-*.yaml
```

由于您要自行创建并管理 worker 机器，因此不需要初始化这些机器。

4. 检查 **<installation_directory>/manifests/cluster-scheduler-02-config.yml** Kubernetes 清单文件中的 **mastersSchedulable** 参数是否已设置为 **false**。此设置可防止在 control plane 机器上调度 pod:

   a. 打开 **<installation_directory>/manifests/cluster-scheduler-02-config.yml** 文件。

   b. 找到 **mastersSchedulable** 参数并确保它被设置为 **false**。

   c. 保存并退出文件。

5. 可选：如果您不希望 Ingress Operator 代表您创建 DNS 记录，请删除 **<installation_directory>/manifests/cluster-dns-02-config.yml** DNS 配置文件中的 **privateZone** 和 **publicZone** 部分：

```
apiVersion: config.openshift.io/v1
kind: DNS
metadata:
  creationTimestamp: null
  name: cluster
spec:
  baseDomain: example.openshift.com
  privateZone: 1
    id: mycluster-100419-private-zone
  publicZone: 2
    id: example.openshift.com
status: {}
```

1 2 完全删除此部分。

如果您这样做，后续步骤中必须手动添加入口 DNS 记录。

6. 要创建 Ignition 配置文件，从包含安装程序的目录运行以下命令：

```
$ ./openshift-install create ignition-configs --dir <installation_directory> 1
```

1 对于 **<installation_directory>**，请指定相同的安装目录。

该目录中将生成以下文件：

```
.
├── auth
│   ├── kubeadmin-password
│   └── kubeconfig
├── bootstrap.ign
├── master.ign
├── metadata.json
└── worker.ign
```

其他资源

- 可选：添加入口 DNS 记录

4.9.6. 导出常用变量

4.9.6.1. 提取基础架构名称

Ignition 配置文件包含一个唯一集群标识符，您可以使用它在 Google Cloud Platform (GCP) 中唯一地标识您的集群。基础架构名称还用于在 OpenShift Container Platform 安装过程中定位适当的 GCP 资源。提供的 Deployment Manager 模板包含对此基础架构名称的引用，因此您必须提取它。

先决条件

- 获取 OpenShift Container Platform 安装程序以及集群的 pull secret。
- 已为集群生成 Ignition 配置文件。
- 安装了 **jq** 软件包。

流程

- 要从 Ignition 配置文件元数据中提取和查看基础架构名称，请运行以下命令：

```
$ jq -r .infraID <installation_directory>/metadata.json 1
```

1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

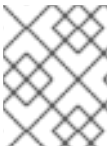
输出示例

```
openshift-vw9j6 1
```

1 此命令的输出是您的集群名称和随机字符串。

4.9.6.2. 为 Deployment Manager 模板导出常用变量

您必须导出与提供的 Deployment Manager 模板搭配使用的一组常用变量，它们有助于在 Google Cloud Platform (GCP) 上完成用户提供基础架构安装。



注意

特定的 Deployment Manager 模板可能还需要其他导出变量，这些变量在相关的程序中详细介绍。

先决条件

- 获取 OpenShift Container Platform 安装程序以及集群的 pull secret。
- 为集群生成 Ignition 配置文件。
- 安装 **jq** 软件包。

流程

1. 导出由提供的 Deployment Manager 模板使用的以下常用变量：

```
$ export BASE_DOMAIN='<base_domain>'
$ export BASE_DOMAIN_ZONE_NAME='<base_domain_zone_name>'
$ export NETWORK_CIDR='10.0.0.0/16'
$ export MASTER_SUBNET_CIDR='10.0.0.0/19'
$ export WORKER_SUBNET_CIDR='10.0.32.0/19'

$ export KUBECONFIG=<installation_directory>/auth/kubeconfig (1)
$ export CLUSTER_NAME=`jq -r .clusterName <installation_directory>/metadata.json`
$ export INFRA_ID=`jq -r .infraID <installation_directory>/metadata.json`
$ export PROJECT_NAME=`jq -r .gcp.projectID <installation_directory>/metadata.json`
$ export REGION=`jq -r .gcp.region <installation_directory>/metadata.json`
```

(1) 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

4.9.7. 在 GCP 中创建 VPC

您必须在 Google Cloud Platform (GCP) 中创建一个 VPC，供您的 OpenShift Container Platform 集群使用。您可以自定义 VPC 来满足您的要求。创建 VPC 的一种方法是修改提供的 Deployment Manager 模板。

注意

如果不使用提供的 Deployment Manager 模板来创建 GCP 基础架构，您必须检查提供的信息并手动创建基础架构。如果集群没有正确初始化，您可能需要联系红帽支持并提供您的安装日志。

先决条件

- 配置 GCP 帐户。
- 为集群生成 Ignition 配置文件。

流程

1. 复制 **VPC 的 Deployment Manager 模板**一节中的模板，并将它以 **01_vpc.py** 形式保存到计算机上。此模板描述了集群所需的 VPC。
2. 创建 **01_vpc.yaml** 资源定义文件：

```
$ cat <<EOF >01_vpc.yaml
imports:
- path: 01_vpc.py

resources:
- name: cluster-vpc
  type: 01_vpc.py
  properties:
    infra_id: '${INFRA_ID}' (1)
    region: '${REGION}' (2)
```

```
    master_subnet_cidr: '${MASTER_SUBNET_CIDR}' 3
    worker_subnet_cidr: '${WORKER_SUBNET_CIDR}' 4
EOF
```

1. **infra_id** 是来自于提取步骤的 **INFRA_ID** 基础架构名称。
2. **region** 是集群要部署到的区域，如 **us-central1**。
3. **master_subnet_cidr** 是 master 子网的 CIDR，如 **10.0.0.0/19**。
4. **worker_subnet_cidr** 是 worker 子网的 CIDR，如 **10.0.32.0/19**。

3. 使用 **gcloud** CLI 创建部署：

```
$ gcloud deployment-manager deployments create ${INFRA_ID}-vpc --config 01_vpc.yaml
```

4.9.7.1. VPC 的 Deployment Manager 模板

您可以使用以下 Deployment Manager 模板来部署 OpenShift Container Platform 集群所需的 VPC：

例 4.1. 01_VPC.py Deployment Manager 模板

```
def GenerateConfig(context):

    resources = [{
        'name': context.properties['infra_id'] + '-network',
        'type': 'compute.v1.network',
        'properties': {
            'region': context.properties['region'],
            'autoCreateSubnetworks': False
        }
    }, {
        'name': context.properties['infra_id'] + '-master-subnet',
        'type': 'compute.v1.subnetwork',
        'properties': {
            'region': context.properties['region'],
            'network': '$(ref.' + context.properties['infra_id'] + '-network.selfLink)',
            'ipCidrRange': context.properties['master_subnet_cidr']
        }
    }, {
        'name': context.properties['infra_id'] + '-worker-subnet',
        'type': 'compute.v1.subnetwork',
        'properties': {
            'region': context.properties['region'],
            'network': '$(ref.' + context.properties['infra_id'] + '-network.selfLink)',
            'ipCidrRange': context.properties['worker_subnet_cidr']
        }
    }, {
        'name': context.properties['infra_id'] + '-router',
        'type': 'compute.v1.router',
        'properties': {
            'region': context.properties['region'],
            'network': '$(ref.' + context.properties['infra_id'] + '-network.selfLink)',
            'nats': [{
```

```
                'name': context.properties['infra_id'] + '-nat-master',
                'natIpAllocateOption': 'AUTO_ONLY',
                'minPortsPerVm': 7168,
                'sourceSubnetworkIpRangesToNat': 'LIST_OF_SUBNETWORKS',
                'subnetworks': [{
                    'name': '$(ref.' + context.properties['infra_id'] + '-master-subnet.selfLink)',
                    'sourceIpRangesToNat': ['ALL_IP_RANGES']
                }]
            }, {
                'name': context.properties['infra_id'] + '-nat-worker',
                'natIpAllocateOption': 'AUTO_ONLY',
                'minPortsPerVm': 512,
                'sourceSubnetworkIpRangesToNat': 'LIST_OF_SUBNETWORKS',
                'subnetworks': [{
                    'name': '$(ref.' + context.properties['infra_id'] + '-worker-subnet.selfLink)',
                    'sourceIpRangesToNat': ['ALL_IP_RANGES']
                }]
            }]
        }
    }]

    return {'resources': resources}
```

4.9.8. 用户置备的基础架构对网络的要求

所有 Red Hat Enterprise Linux CoreOS（RHCOS）机器在启动过程中需要 **initramfs** 中的网络从机器配置服务器获取 Ignition 配置。

您必须配置机器间的网络连接，以便集群组件进行通信。每台机器都必须能够解析集群中所有其他机器的主机名。

表 4.32. 所有机器到所有机器

协议	端口	描述
ICMP	N/A	网络可访问性测试
TCP	**1936**	指标
	9000-9999	主机级别的服务，包括端口 **9100-9101** 上的节点导出器和端口 **9099** 上的 Cluster Version Operator。
	10250-10259	Kubernetes 保留的默认端口
	10256	openshift-sdn
UDP	**4789**	VXLAN 和 Geneve
	6081	VXLAN 和 Geneve
	9000-9999	主机级别的服务，包括端口 **9100-9101** 上的节点导出器。

协议	端口	描述
TCP/UDP	**30000-32767**	Kubernetes 节点端口

表 4.33. 要通过控制平面的所有机器

协议	端口	描述
TCP	**6443**	Kubernetes API

表 4.34. control plane 机器到 control plane 机器

协议	端口	描述
TCP	**2379-2380**	etcd 服务器和对等端口

网络拓扑要求

您为集群置备的基础架构必须满足下列网络拓扑要求。

重要

OpenShift Container Platform 要求所有节点都能访问互联网，以便为平台容器提取镜像并向红帽提供遥测数据。

负载均衡器

在安装 OpenShift Container Platform 前，您必须置备两个满足以下要求的负载均衡器：

1. **API 负载均衡器**：提供一个通用端点，供用户（包括人和机器）与平台交互和配置。配置以下条件：
   - 只适用于第 4 层负载均衡。这可被称为 Raw TCP、SSL Passthrough 或者 SSL 桥接模式。如果使用 SSL Bridge 模式，必须为 API 路由启用 Server Name Indication（SNI）。
   - 无状态负载平衡算法。这些选项根据负载均衡器的实现而有所不同。

   重要

   不要为 API 负载均衡器配置会话持久性。

   在负载均衡器的前端和后台配置以下端口：

   表 4.35. API 负载均衡器

端口	后端机器（池成员）	内部	外部	描述

端口	后端机器（池成员）	内部	外部	描述
6443	Bootstrap 和 control plane.bootstrap 机器初始化集群 control plane 后，您要从负载均衡器中删除 bootstrap 机器。您必须为 API 服务器健康检查探测配置 **/readyz** 端点。	X	X	Kubernetes API 服务器
22623	Bootstrap 和 control plane.bootstrap 机器初始化集群 control plane 后，您要从负载均衡器中删除 bootstrap 机器。	X		机器配置服务器

注意

负载均衡器必须配置为，从 API 服务器关闭 **/readyz** 端点到从池中删除 API 服务器实例时最多需要 30 秒。在 **/readyz** 返回错误或处于健康状态后的时间范围内，端点必须被删除或添加。每 5 秒或 10 秒探测一次，有两个成功请求处于健康状态，三个成为不健康的请求经过测试。

2. **应用程序入口负载均衡器**:提供来自集群外部的应用程序流量流量的 Ingress 点。配置以下条件：

   - 只适用于第 4 层负载均衡。这可被称为 Raw TCP、SSL Passthrough 或者 SSL 桥接模式。如果使用 SSL Bridge 模式，您必须为 Ingress 路由启用Server Name Indication（SNI）。
   - 建议根据可用选项以及平台上托管的应用程序类型，使用基于连接的或者基于会话的持久性。

   在负载均衡器的前端和后台配置以下端口：

表 4.36. 应用程序入口负载均衡器

端口	后端机器（池成员）	内部	外部	描述
443	默认运行入口路由器 Pod、计算或 worker 的机器。	X	X	HTTPS 流量
80	默认运行入口路由器 Pod、计算或 worker 的机器。	X	X	HTTP 流量

提示

如果负载均衡器可以看到客户端的真实 IP 地址，启用基于 IP 的会话持久性可提高使用端到端 TLS 加密的应用程序的性能。

注意

OpenShift Container Platform 集群需要正确配置入口路由器。control plane 初始化后，您必须配置入口路由器。

4.9.9. 在 GCP 中创建负载均衡器

您必须在 Google Cloud Platform (GCP) 中配置负载均衡器，供您的 OpenShift Container Platform 集群使用。创建这些组件的一种方法是修改提供的 Deployment Manager 模板。

注意

如果不使用提供的 Deployment Manager 模板来创建 GCP 基础架构，您必须检查提供的信息并手动创建基础架构。如果集群没有正确初始化，您可能需要联系红帽支持并提供您的安装日志。

先决条件

- 配置 GCP 帐户。
- 为集群生成 Ignition 配置文件。
- 在 GCP 中创建和配置 VPC 及相关子网。

流程

1. 复制**负载均衡器的 Deployment Manager 模板**一节中的模板，并将它以 **02_lb_int.py** 形式保存到计算机上。此模板描述了集群所需的负载均衡对象。
2. 对于外部集群，还要将本主题的 **外部负载均衡器的 Deployment Manager 模板** 部分中的模板进行复制，并将它以 **02_lb_ext.py** 形式保存到计算机上。此模板描述了集群所需的外部负载均衡对象。
3. 导出部署模板使用的变量：
   a. 导出集群网络位置：

```
$ export CLUSTER_NETWORK=(`gcloud compute networks describe ${INFRA_ID}-network --format json | jq -r .selfLink`)
```

   b. 导出 control plane 子网位置：

```
$ export CONTROL_SUBNET=(`gcloud compute networks subnets describe ${INFRA_ID}-master-subnet --region=${REGION} --format json | jq -r .selfLink`)
```

   c. 导出集群使用的三个区域（zone）：

```
$ export ZONE_0=(`gcloud compute regions describe ${REGION} --format=json | jq -r .zones[0] | cut -d "/" -f9`)
```

```
$ export ZONE_1=(`gcloud compute regions describe ${REGION} --format=json | jq -r .zones[1] | cut -d "/" -f9`)
```

```
$ export ZONE_2=(`gcloud compute regions describe ${REGION} --format=json | jq -r .zones[2] | cut -d "/" -f9`)
```

4. 创建 **02_infra.yaml** 资源定义文件：

```
$ cat <<EOF >02_infra.yaml
imports:
- path: 02_lb_ext.py
- path: 02_lb_int.py 1
resources:
- name: cluster-lb-ext 2
  type: 02_lb_ext.py
  properties:
    infra_id: '${INFRA_ID}' 3
    region: '${REGION}' 4
- name: cluster-lb-int
  type: 02_lb_int.py
  properties:
    cluster_network: '${CLUSTER_NETWORK}'
    control_subnet: '${CONTROL_SUBNET}' 5
    infra_id: '${INFRA_ID}'
    region: '${REGION}'
    zones: 6
    - '${ZONE_0}'
    - '${ZONE_1}'
    - '${ZONE_2}'
EOF
```

1 2 仅在部署外部集群时需要。

3 **infra_id** 是来自于提取步骤的 **INFRA_ID** 基础架构名称。

4 **region** 是集群要部署到的区域，如 **us-central1**。

5 **control_subnet** 是到控制子网的 URI。

6 **zones** 是 control plane 实例要部署到的区域，如 **us-east1-b**、**us-east1-c** 和 **us-east1-d**。

5. 使用 **gcloud** CLI 创建部署：

```
$ gcloud deployment-manager deployments create ${INFRA_ID}-infra --config 02_infra.yaml
```

6. 导出集群 IP 地址：

```
$ export CLUSTER_IP=(`gcloud compute addresses describe ${INFRA_ID}-cluster-ip --
region=${REGION} --format json | jq -r .address`)
```

7. 对于外部集群，还要导出集群公共 IP 地址：

```
$ export CLUSTER_PUBLIC_IP=(`gcloud compute addresses describe ${INFRA_ID}-cluster-
public-ip --region=${REGION} --format json | jq -r .address`)
```

4.9.9.1. 外部负载均衡器的 Deployment Manager 模板

您可以使用以下 Deployment Manager 模板来部署 OpenShift Container Platform 集群所需的外部负载均衡器：

例 4.2. 02_lb_ext.py Deployment Manager 模板

```
def GenerateConfig(context):

    resources = [{
        'name': context.properties['infra_id'] + '-cluster-public-ip',
        'type': 'compute.v1.address',
        'properties': {
            'region': context.properties['region']
        }
    }, {
        # Refer to docs/dev/kube-apiserver-health-check.md on how to correctly setup health check
probe for kube-apiserver
        'name': context.properties['infra_id'] + '-api-http-health-check',
        'type': 'compute.v1.httpHealthCheck',
        'properties': {
            'port': 6080,
            'requestPath': '/readyz'
        }
    }, {
        'name': context.properties['infra_id'] + '-api-target-pool',
        'type': 'compute.v1.targetPool',
        'properties': {
            'region': context.properties['region'],
            'healthChecks': ['$(ref.' + context.properties['infra_id'] + '-api-http-health-check.selfLink)'],
            'instances': []
        }
    }, {
        'name': context.properties['infra_id'] + '-api-forwarding-rule',
        'type': 'compute.v1.forwardingRule',
        'properties': {
            'region': context.properties['region'],
            'IPAddress': '$(ref.' + context.properties['infra_id'] + '-cluster-public-ip.selfLink)',
            'target': '$(ref.' + context.properties['infra_id'] + '-api-target-pool.selfLink)',
            'portRange': '6443'
        }
    }]

    return {'resources': resources}
```

4.9.9.2. 内部负载均衡器的 Deployment Manager 模板

您可以使用以下 Deployment Manager 模板来部署 OpenShift Container Platform 集群所需的内部负载均衡器：

例 4.3. 02_lb_int.py Deployment Manager 模板

```
def GenerateConfig(context):

    backends = []
    for zone in context.properties['zones']:
        backends.append({
            'group': '$(ref.' + context.properties['infra_id'] + '-master-' + zone + '-instance-group' +
```

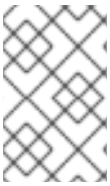
```
'.selfLink)'
        })

    resources = [{
        'name': context.properties['infra_id'] + '-cluster-ip',
        'type': 'compute.v1.address',
        'properties': {
            'addressType': 'INTERNAL',
            'region': context.properties['region'],
            'subnetwork': context.properties['control_subnet']
        }
    }, {
        # Refer to docs/dev/kube-apiserver-health-check.md on how to correctly setup health check
probe for kube-apiserver
        'name': context.properties['infra_id'] + '-api-internal-health-check',
        'type': 'compute.v1.healthCheck',
        'properties': {
            'httpsHealthCheck': {
                'port': 6443,
                'requestPath': '/readyz'
            },
            'type': "HTTPS"
        }
    }, {
        'name': context.properties['infra_id'] + '-api-internal-backend-service',
        'type': 'compute.v1.regionBackendService',
        'properties': {
            'backends': backends,
            'healthChecks': ['$(ref.' + context.properties['infra_id'] + '-api-internal-health-
check.selfLink)'],
            'loadBalancingScheme': 'INTERNAL',
            'region': context.properties['region'],
            'protocol': 'TCP',
            'timeoutSec': 120
        }
    }, {
        'name': context.properties['infra_id'] + '-api-internal-forwarding-rule',
        'type': 'compute.v1.forwardingRule',
        'properties': {
            'backendService': '$(ref.' + context.properties['infra_id'] + '-api-internal-backend-
service.selfLink)',
            'IPAddress': '$(ref.' + context.properties['infra_id'] + '-cluster-ip.selfLink)',
            'loadBalancingScheme': 'INTERNAL',
            'ports': ['6443','22623'],
            'region': context.properties['region'],
            'subnetwork': context.properties['control_subnet']
        }
    }]

    for zone in context.properties['zones']:
        resources.append({
            'name': context.properties['infra_id'] + '-master-' + zone + '-instance-group',
            'type': 'compute.v1.instanceGroup',
            'properties': {
                'namedPorts': [
                    {
```

```
                'name': 'ignition',
                'port': 22623
            }, {
                'name': 'https',
                'port': 6443
            }
        ],
        'network': context.properties['cluster_network'],
        'zone': zone
    }
})

return {'resources': resources}
```

创建外部集群时，除了 **02_lb_ext.py** 模板外，还需要此模板。

4.9.10. 在 GCP 中创建私有 DNS 区域

您必须在 Google Cloud Platform (GCP) 中配置一个私人的 DNS 区以供您的 OpenShift Container Platform 集群使用。创建此组件的一种方法是修改提供的 Deployment Manager 模板。



注意

如果不使用提供的 Deployment Manager 模板来创建 GCP 基础架构，您必须检查提供的信息并手动创建基础架构。如果集群没有正确初始化，您可能需要联系红帽支持并提供您的安装日志。

先决条件

- 配置 GCP 帐户。
- 为集群生成 Ignition 配置文件。
- 在 GCP 中创建和配置 VPC 及相关子网。

流程

1. 复制 **私人 DNS 的 Deployment Manager 模板**一节中的模板，并将它以 **02_dns.py** 形式保存到计算机上。此模板描述了集群所需的私有 DNS 对象。
2. 创建 **02_dns.yaml** 资源定义文件：

```
$ cat <<EOF >02_dns.yaml
imports:
- path: 02_dns.py

resources:
- name: cluster-dns
  type: 02_dns.py
  properties:
    infra_id: '${INFRA_ID}' (1)
```

```
  cluster_domain: '${CLUSTER_NAME}.${BASE_DOMAIN}' (2)
  cluster_network: '${CLUSTER_NETWORK}' (3)
EOF
```

1. **infra_id** 是来自于提取步骤的 **INFRA_ID** 基础架构名称。
2. **cluster_domain** 是集群的域，如 **openshift.example.com**。
3. **cluster_network** 是集群网络的 **selfLink** URL。

3. 使用 **gcloud** CLI 创建部署：

```
$ gcloud deployment-manager deployments create ${INFRA_ID}-dns --config 02_dns.yaml
```

4. 由于 Deployment Manager 的限制，模板不会创建 DNS 条目，因此您必须手动创建它们：

   a. 添加内部 DNS 条目：

```
$ if [ -f transaction.yaml ]; then rm transaction.yaml; fi
$ gcloud dns record-sets transaction start --zone ${INFRA_ID}-private-zone
$ gcloud dns record-sets transaction add ${CLUSTER_IP} --name
api.${CLUSTER_NAME}.${BASE_DOMAIN}. --ttl 60 --type A --zone ${INFRA_ID}-
private-zone
$ gcloud dns record-sets transaction add ${CLUSTER_IP} --name api-
int.${CLUSTER_NAME}.${BASE_DOMAIN}. --ttl 60 --type A --zone ${INFRA_ID}-
private-zone
$ gcloud dns record-sets transaction execute --zone ${INFRA_ID}-private-zone
```

   b. 对于外部集群，还要添加外部 DNS 条目：

```
$ if [ -f transaction.yaml ]; then rm transaction.yaml; fi
$ gcloud dns record-sets transaction start --zone ${BASE_DOMAIN_ZONE_NAME}
$ gcloud dns record-sets transaction add ${CLUSTER_PUBLIC_IP} --name
api.${CLUSTER_NAME}.${BASE_DOMAIN}. --ttl 60 --type A --zone
${BASE_DOMAIN_ZONE_NAME}
$ gcloud dns record-sets transaction execute --zone ${BASE_DOMAIN_ZONE_NAME}
```

4.9.10.1. 私有 DNS 的 Deployment Manager 模板

您可以使用以下 Deployment Manager 模板来部署 OpenShift Container Platform 集群所需的 私有 DNS：

例 4.4. 02_dns.py Deployment Manager 模板

```
def GenerateConfig(context):

  resources = [{
    'name': context.properties['infra_id'] + '-private-zone',
    'type': 'dns.v1.managedZone',
    'properties': {
      'description': '',
      'dnsName': context.properties['cluster_domain'] + '.',
      'visibility': 'private',
```

```
            'privateVisibilityConfig': {
               'networks': [{
                  'networkUrl': context.properties['cluster_network']
               }]
            }
         }
      }]

      return {'resources': resources}
```

4.9.11. 在 GCP 中创建防火墙规则

您必须在 Google Cloud Platform (GCP) 中创建一个防火墙，供您的 OpenShift Container Platform 集群使用。创建这些组件的一种方法是修改提供的 Deployment Manager 模板。

注意

如果不使用提供的 Deployment Manager 模板来创建 GCP 基础架构，您必须检查提供的信息并手动创建基础架构。如果集群没有正确初始化，您可能需要联系红帽支持并提供您的安装日志。

先决条件

- 配置 GCP 帐户。
- 为集群生成 Ignition 配置文件。
- 在 GCP 中创建和配置 VPC 及相关子网。

流程

1. 复制**防火墙规则的 Deployment Manager 模板**一节中的模板，并将它以 **03_firewall.py** 形式保存到计算机上。此模板描述了集群所需的安全组。
2. 创建 **03_firewall.yaml** 资源定义文件：

```
$ cat <<EOF >03_firewall.yaml
imports:
- path: 03_firewall.py

resources:
- name: cluster-firewall
  type: 03_firewall.py
  properties:
    allowed_external_cidr: '0.0.0.0/0' (1)
    infra_id: '${INFRA_ID}' (2)
    cluster_network: '${CLUSTER_NETWORK}' (3)
    network_cidr: '${NETWORK_CIDR}' (4)
EOF
```

(1) **allowed_external_cidr** 是 CIDR 范围，可访问集群 API 和 SSH 到 bootstrap 主机。对于内部集群，将此值设置为 **${NETWORK_CIDR}**。

2. **infra_id** 是来自于提取步骤的 **INFRA_ID** 基础架构名称。
3. **cluster_network** 是集群网络的 **selfLink** URL。
4. **network_cidr** 是 VPC 网络的 CIDR，如 **10.0.0.0/16**。

3. 使用 **gcloud** CLI 创建部署：

```
$ gcloud deployment-manager deployments create ${INFRA_ID}-firewall --config
03_firewall.yaml
```

4.9.11.1. 防火墙规则的 Deployment Manager 模板

您可以使用以下 Deployment Manager 模板来部署 OpenShift Container Platform 集群所需的防火墙规则：

例 4.5. 03_firewall.py Deployment Manager 模板

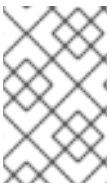
```
def GenerateConfig(context):

    resources = [{
        'name': context.properties['infra_id'] + '-bootstrap-in-ssh',
        'type': 'compute.v1.firewall',
        'properties': {
            'network': context.properties['cluster_network'],
            'allowed': [{
                'IPProtocol': 'tcp',
                'ports': ['22']
            }],
            'sourceRanges': [context.properties['allowed_external_cidr']],
            'targetTags': [context.properties['infra_id'] + '-bootstrap']
        }
    }, {
        'name': context.properties['infra_id'] + '-api',
        'type': 'compute.v1.firewall',
        'properties': {
            'network': context.properties['cluster_network'],
            'allowed': [{
                'IPProtocol': 'tcp',
                'ports': ['6443']
            }],
            'sourceRanges': [context.properties['allowed_external_cidr']],
            'targetTags': [context.properties['infra_id'] + '-master']
        }
    }, {
        'name': context.properties['infra_id'] + '-health-checks',
        'type': 'compute.v1.firewall',
        'properties': {
            'network': context.properties['cluster_network'],
            'allowed': [{
                'IPProtocol': 'tcp',
                'ports': ['6080', '6443', '22624']
            }],
            'sourceRanges': ['35.191.0.0/16', '130.211.0.0/22', '209.85.152.0/22', '209.85.204.0/22'],
```

```
            'targetTags': [context.properties['infra_id'] + '-master']
        }
    }, {
        'name': context.properties['infra_id'] + '-etcd',
        'type': 'compute.v1.firewall',
        'properties': {
            'network': context.properties['cluster_network'],
            'allowed': [{
                'IPProtocol': 'tcp',
                'ports': ['2379-2380']
            }],
            'sourceTags': [context.properties['infra_id'] + '-master'],
            'targetTags': [context.properties['infra_id'] + '-master']
        }
    }, {
        'name': context.properties['infra_id'] + '-control-plane',
        'type': 'compute.v1.firewall',
        'properties': {
            'network': context.properties['cluster_network'],
            'allowed': [{
                'IPProtocol': 'tcp',
                'ports': ['10257']
            },{
                'IPProtocol': 'tcp',
                'ports': ['10259']
            },{
                'IPProtocol': 'tcp',
                'ports': ['22623']
            }],
            'sourceTags': [
                context.properties['infra_id'] + '-master',
                context.properties['infra_id'] + '-worker'
            ],
            'targetTags': [context.properties['infra_id'] + '-master']
        }
    }, {
        'name': context.properties['infra_id'] + '-internal-network',
        'type': 'compute.v1.firewall',
        'properties': {
            'network': context.properties['cluster_network'],
            'allowed': [{
                'IPProtocol': 'icmp'
            },{
                'IPProtocol': 'tcp',
                'ports': ['22']
            }],
            'sourceRanges': [context.properties['network_cidr']],
            'targetTags': [
                context.properties['infra_id'] + '-master',
                context.properties['infra_id'] + '-worker'
            ]
        }
    }, {
        'name': context.properties['infra_id'] + '-internal-cluster',
        'type': 'compute.v1.firewall',
        'properties': {
```

```
            'network': context.properties['cluster_network'],
            'allowed': [{
                'IPProtocol': 'udp',
                'ports': ['4789', '6081']
            },{
                'IPProtocol': 'tcp',
                'ports': ['9000-9999']
            },{
                'IPProtocol': 'udp',
                'ports': ['9000-9999']
            },{
                'IPProtocol': 'tcp',
                'ports': ['10250']
            },{
                'IPProtocol': 'tcp',
                'ports': ['30000-32767']
            },{
                'IPProtocol': 'udp',
                'ports': ['30000-32767']
            }],
            'sourceTags': [
                context.properties['infra_id'] + '-master',
                context.properties['infra_id'] + '-worker'
            ],
            'targetTags': [
                context.properties['infra_id'] + '-master',
                context.properties['infra_id'] + '-worker'
            ]
        }
    }]

    return {'resources': resources}
```

4.9.12. 在 GCP 中创建 IAM 角色

您必须在 Google Cloud Platform (GCP) 中创建一个 IAM 角色，供您的 OpenShift Container Platform 集群使用。创建这些组件的一种方法是修改提供的 Deployment Manager 模板。



注意

如果不使用提供的 Deployment Manager 模板来创建 GCP 基础架构，您必须检查提供的信息并手动创建基础架构。如果集群没有正确初始化，您可能需要联系红帽支持并提供您的安装日志。

先决条件

- 配置 GCP 帐户。
- 为集群生成 Ignition 配置文件。
- 在 GCP 中创建和配置 VPC 及相关子网。

流程

1. 复制 **IAM 角色的 Deployment Manager 模板**一节中的模板，并将它以 **03_iam.p** 形式保存到计算机上。此模板描述了集群所需的 IAM 角色。

2. 创建 **03_iam.yaml** 资源定义文件：

```
$ cat <<EOF >03_iam.yaml
imports:
- path: 03_iam.py
resources:
- name: cluster-iam
  type: 03_iam.py
  properties:
    infra_id: '${INFRA_ID}' (1)
EOF
```

(1) **infra_id** 是来自于提取步骤的 **INFRA_ID** 基础架构名称。

3. 使用 **gcloud** CLI 创建部署：

```
$ gcloud deployment-manager deployments create ${INFRA_ID}-iam --config 03_iam.yaml
```

4. 导出 master 服务帐户的变量：

```
$ export MASTER_SERVICE_ACCOUNT=(`gcloud iam service-accounts list --filter
"email~^${INFRA_ID}-m@${PROJECT_NAME}." --format json | jq -r '.[0].email'`)
```

5. 导出 worker 服务帐户的变量：

```
$ export WORKER_SERVICE_ACCOUNT=(`gcloud iam service-accounts list --filter
"email~^${INFRA_ID}-w@${PROJECT_NAME}." --format json | jq -r '.[0].email'`)
```

6. 导出托管计算机器的子网的变量：

```
$ export COMPUTE_SUBNET=(`gcloud compute networks subnets describe ${INFRA_ID}-
worker-subnet --region=${REGION} --format json | jq -r .selfLink`)
```

7. 由于 Deployment Manager 的限制，模板不会创建策略绑定，因此您必须手动创建它们：

```
$ gcloud projects add-iam-policy-binding ${PROJECT_NAME} --member
"serviceAccount:${MASTER_SERVICE_ACCOUNT}" --role "roles/compute.instanceAdmin"
$ gcloud projects add-iam-policy-binding ${PROJECT_NAME} --member
"serviceAccount:${MASTER_SERVICE_ACCOUNT}" --role "roles/compute.networkAdmin"
$ gcloud projects add-iam-policy-binding ${PROJECT_NAME} --member
"serviceAccount:${MASTER_SERVICE_ACCOUNT}" --role "roles/compute.securityAdmin"
$ gcloud projects add-iam-policy-binding ${PROJECT_NAME} --member
"serviceAccount:${MASTER_SERVICE_ACCOUNT}" --role "roles/iam.serviceAccountUser"
$ gcloud projects add-iam-policy-binding ${PROJECT_NAME} --member
"serviceAccount:${MASTER_SERVICE_ACCOUNT}" --role "roles/storage.admin"

$ gcloud projects add-iam-policy-binding ${PROJECT_NAME} --member
"serviceAccount:${WORKER_SERVICE_ACCOUNT}" --role "roles/compute.viewer"
$ gcloud projects add-iam-policy-binding ${PROJECT_NAME} --member
"serviceAccount:${WORKER_SERVICE_ACCOUNT}" --role "roles/storage.admin"
```

8. 创建服务帐户密钥，并将它保存到本地备用：

```
$ gcloud iam service-accounts keys create service-account-key.json --iam-
account=${MASTER_SERVICE_ACCOUNT}
```

4.9.12.1. IAM 角色的 Deployment Manager 模板

您可以使用以下 Deployment Manager 模板来部署 OpenShift Container Platform 集群所需的 IAM 角色：

例 4.6. 03_iam.py Deployment Manager 模板

```
def GenerateConfig(context):

    resources = [{
        'name': context.properties['infra_id'] + '-master-node-sa',
        'type': 'iam.v1.serviceAccount',
        'properties': {
            'accountId': context.properties['infra_id'] + '-m',
            'displayName': context.properties['infra_id'] + '-master-node'
        }
    }, {
        'name': context.properties['infra_id'] + '-worker-node-sa',
        'type': 'iam.v1.serviceAccount',
        'properties': {
            'accountId': context.properties['infra_id'] + '-w',
            'displayName': context.properties['infra_id'] + '-worker-node'
        }
    }]

    return {'resources': resources}
```

4.9.13. 为 GCP 基础架构创建 RHCOS 集群镜像

您必须对 OpenShift Container Platform 节点的 Google Cloud Platform (GCP) 使用有效的 Red Hat Enterprise Linux CoreOS (RHCOS) 镜像。

流程

1. 从 RHCOS 镜像镜像页面获取 RHCOS 镜像。

重要

RHCOS 镜像可能不会随着 OpenShift Container Platform 的每一发行版本都有改变。您必须下载一个最高版本的镜像，其版本号应小于或等于您安装的 OpenShift Container Platform 版本。如果可用，请使用与 OpenShift Container Platform 版本匹配的镜像版本。

文件名包含 OpenShift Container Platform 版本号，格式为 **rhcos-<version>-<arch>-gcp.<arch>.tar.gz**。

2. 创建 Google 存储桶：

```
$ gsutil mb gs://<bucket_name>
```

3. 将 RHCOS 镜像上传到 Google 存储桶：

```
$ gsutil cp <downloaded_image_file_path>/rhcos-<version>-x86_64-gcp.x86_64.tar.gz
gs://<bucket_name>
```

4. 将上传的 RHCOS 镜像位置导出为变量：

```
$ export IMAGE_SOURCE="gs://<bucket_name>/rhcos-<version>-x86_64-
gcp.x86_64.tar.gz"
```

5. 创建集群镜像：

```
$ gcloud compute images create "${INFRA_ID}-rhcos-image" \
    --source-uri="${IMAGE_SOURCE}"
```

4.9.14. 在 GCP 中创建 bootstrap 机器

您必须在 Google Cloud Platform (GCP) 中创建 bootstrap 机器，以便在 OpenShift Container Platform 集群初始化过程中使用。创建此机器的一种方法是修改提供的 Deployment Manager 模板。

注意

如果不使用提供的 Deployment Manager 模板来创建 bootstrap 机器，您必须检查提供的信息并手动创建基础架构。如果集群没有正确初始化，您可能需要联系红帽支持并提供您的安装日志。

先决条件

- 配置 GCP 帐户。
- 为集群生成 Ignition 配置文件。
- 在 GCP 中创建和配置 VPC 及相关子网。
- 在 GCP 中创建和配置联网及负载均衡器。
- 创建 control plane 和计算角色。
- 确定安装了 pyOpenSSL。

流程

1. 复制 **bootstrap 机器的 Deployment Manager 模板**一节中的模板，并将它以 **04_bootstrap.py** 形式保存到计算机上。此模板描述了集群所需的 bootstrap 机器。
2. 导出安装程序所需的 Red Hat Enterprise Linux CoreOS（RHCOS）镜像的位置：

```
$ export CLUSTER_IMAGE=(`gcloud compute images describe ${INFRA_ID}-rhcos-image --
format json | jq -r .selfLink`)
```

3. 创建存储桶并上传 **bootstrap.ign** 文件：

```
$ gsutil mb gs://${INFRA_ID}-bootstrap-ignition
$ gsutil cp <installation_directory>/bootstrap.ign gs://${INFRA_ID}-bootstrap-ignition/
```

4. 为 bootstrap 实例创建一个签名的 URL，用于访问 Ignition 配置。将输出中的 URL 导出为变量：

```
$ export BOOTSTRAP_IGN=`gsutil signurl -d 1h service-account-key.json gs://${INFRA_ID}-bootstrap-ignition/bootstrap.ign | grep "^gs:" | awk '{print $5}'`
```

5. 创建 **04_bootstrap.yaml** 资源定义文件：

```
$ cat <<EOF >04_bootstrap.yaml
imports:
- path: 04_bootstrap.py

resources:
- name: cluster-bootstrap
  type: 04_bootstrap.py
  properties:
    infra_id: '${INFRA_ID}' (1)
    region: '${REGION}' (2)
    zone: '${ZONE_0}' (3)

    cluster_network: '${CLUSTER_NETWORK}' (4)
    control_subnet: '${CONTROL_SUBNET}' (5)
    image: '${CLUSTER_IMAGE}' (6)
    machine_type: 'n1-standard-4' (7)
    root_volume_size: '128' (8)

    bootstrap_ign: '${BOOTSTRAP_IGN}' (9)
EOF
```

(1) **infra_id** 是来自于提取步骤的 **INFRA_ID** 基础架构名称。

(2) **region** 是集群要部署到的区域，如 **us-central1**。

(3) **zone** 是 Bootstrap 实例要部署到的区域，如 **us-central1-b**。

(4) **cluster_network** 是集群网络的 **selfLink** URL。

(5) **control_subnet** 是控制子网的 **selfLink** URL。

(6) **image** 是 RHCOS 镜像的 **selfLink** URL。

(7) **machine_type** 是实例的机器类型，如 **n1-standard-4**。

(8) **root_volume_size** 是 bootstrap 机器的引导磁盘大小。

(9) **bootstrap_ign** 是创建签名 URL 时的 URL 输出。

6. 使用 **gcloud** CLI 创建部署：

```
$ gcloud deployment-manager deployments create ${INFRA_ID}-bootstrap --config 04_bootstrap.yaml
```

7. 由于 Deployment Manager 的限制，模板无法管理负载均衡器成员资格，因此您必须手动添加 bootstrap 机器：

   a. 将 bootstrap 实例添加到内部负载均衡器实例组中：

   ```
   $ gcloud compute instance-groups unmanaged add-instances \
     ${INFRA_ID}-bootstrap-instance-group --zone=${ZONE_0} --instances=${INFRA_ID}-
   bootstrap
   ```

   b. 将 bootstrap 实例组添加到内部负载均衡器后端服务中：

   ```
   $ gcloud compute backend-services add-backend \
     ${INFRA_ID}-api-internal-backend-service --region=${REGION} --instance-
   group=${INFRA_ID}-bootstrap-instance-group --instance-group-zone=${ZONE_0}
   ```

4.9.14.1. bootstrap 机器的 Deployment Manager 模板

您可以使用以下 Deployment Manager 模板来部署 OpenShift Container Platform 集群所需的 bootstrap 机器：

例 4.7. 04_bootstrap.py Deployment Manager 模板

```
def GenerateConfig(context):

    resources = [{
        'name': context.properties['infra_id'] + '-bootstrap-public-ip',
        'type': 'compute.v1.address',
        'properties': {
            'region': context.properties['region']
        }
    }, {
        'name': context.properties['infra_id'] + '-bootstrap',
        'type': 'compute.v1.instance',
        'properties': {
            'disks': [{
                'autoDelete': True,
                'boot': True,
                'initializeParams': {
                    'diskSizeGb': context.properties['root_volume_size'],
                    'sourceImage': context.properties['image']
                }
            }],
            'machineType': 'zones/' + context.properties['zone'] + '/machineTypes/' +
context.properties['machine_type'],
            'metadata': {
                'items': [{
                    'key': 'user-data',
                    'value': '{"ignition":{"config":{"replace":{"source":"' + context.properties['bootstrap_ign']
+ '"}},"version":"3.1.0"}}',
                }]
            },
            'networkInterfaces': [{
                'subnetwork': context.properties['control_subnet'],
                'accessConfigs': [{
                    'natIP': '$(ref.' + context.properties['infra_id'] + '-bootstrap-public-ip.address)'
```

```
            }]
        }],
        'tags': {
            'items': [
                context.properties['infra_id'] + '-master',
                context.properties['infra_id'] + '-bootstrap'
            ]
        },
        'zone': context.properties['zone']
    }
}, {
    'name': context.properties['infra_id'] + '-bootstrap-instance-group',
    'type': 'compute.v1.instanceGroup',
    'properties': {
        'namedPorts': [
            {
                'name': 'ignition',
                'port': 22623
            }, {
                'name': 'https',
                'port': 6443
            }
        ],
        'network': context.properties['cluster_network'],
        'zone': context.properties['zone']
    }
}]

return {'resources': resources}
```

4.9.15. 在 GCP 中创建 control plane 机器

您必须在 Google Cloud Platform (GCP) 中创建 control plane 机器，供您的集群使用。创建这些机器的一种方法是修改提供的 Deployment Manager 模板。

注意

如果不使用提供的 Deployment Manager 模板来创建 control plane 机器，您必须检查提供的信息并手动创建基础架构。如果集群没有正确初始化，您可能需要联系红帽支持并提供您的安装日志。

先决条件

- 配置 GCP 帐户。
- 为集群生成 Ignition 配置文件。
- 在 GCP 中创建和配置 VPC 及相关子网。
- 在 GCP 中创建和配置联网及负载均衡器。
- 创建 control plane 和计算角色。
- 创建 bootstrap 机器。

流程

1. 复制 **control plane 机器的 Deployment Manager 模板**一节中的模板，并将它以 **05_control_plane.py** 形式保存到计算机上。此模板描述了集群所需的 control plane 机器。
2. 导出资源定义所需的以下变量：

```
$ export MASTER_IGNITION=`cat <installation_directory>/master.ign`
```

3. 创建 **05_control_plane.yaml** 资源定义文件：

```
$ cat <<EOF >05_control_plane.yaml
imports:
- path: 05_control_plane.py

resources:
- name: cluster-control-plane
  type: 05_control_plane.py
  properties:
    infra_id: '${INFRA_ID}' (1)
    zones: (2)
    - '${ZONE_0}'
    - '${ZONE_1}'
    - '${ZONE_2}'

    control_subnet: '${CONTROL_SUBNET}' (3)
    image: '${CLUSTER_IMAGE}' (4)
    machine_type: 'n1-standard-4' (5)
    root_volume_size: '128'
    service_account_email: '${MASTER_SERVICE_ACCOUNT}' (6)

    ignition: '${MASTER_IGNITION}' (7)
EOF
```

(1) **infra_id** 是来自于提取步骤的 **INFRA_ID** 基础架构名称。

(2) **zones** 是 control plane 实例要部署到的区域，如 **us-central1-a**、**us-central1-b** 和 **us-central1-c**。

(3) **control_subnet** 是控制子网的 **selfLink** URL。

(4) **image** 是 RHCOS 镜像的 **selfLink** URL。

(5) **machine_type** 是实例的机器类型，如 **n1-standard-4**。

(6) **service_account_email** 是创建的 master 服务帐户的电子邮件地址。

(7) **ignition** 是 **master.ign** 文件的内容。

4. 使用 **gcloud** CLI 创建部署：

```
$ gcloud deployment-manager deployments create ${INFRA_ID}-control-plane --config
05_control_plane.yaml
```

5. 由于 Deployment Manager 的限制，模板无法管理负载均衡器成员资格，因此您必须手动添加 control plane 机器。

   - 运行以下命令，将 control plane 机器添加到适当的实例组中：

     ```
     $ gcloud compute instance-groups unmanaged add-instances ${INFRA_ID}-master-${ZONE_0}-instance-group --zone=${ZONE_0} --instances=${INFRA_ID}-master-0
     $ gcloud compute instance-groups unmanaged add-instances ${INFRA_ID}-master-${ZONE_1}-instance-group --zone=${ZONE_1} --instances=${INFRA_ID}-master-1
     $ gcloud compute instance-groups unmanaged add-instances ${INFRA_ID}-master-${ZONE_2}-instance-group --zone=${ZONE_2} --instances=${INFRA_ID}-master-2
     ```

   - 对于外部集群，还需要运行以下命令将 control plane 机器添加到目标池中：

     ```
     $ gcloud compute target-pools add-instances ${INFRA_ID}-api-target-pool --instances-zone="${ZONE_0}" --instances=${INFRA_ID}-master-0
     $ gcloud compute target-pools add-instances ${INFRA_ID}-api-target-pool --instances-zone="${ZONE_1}" --instances=${INFRA_ID}-master-1
     $ gcloud compute target-pools add-instances ${INFRA_ID}-api-target-pool --instances-zone="${ZONE_2}" --instances=${INFRA_ID}-master-2
     ```

4.9.15.1. control plane 机器的 Deployment Manager 模板

您可以使用以下 Deployment Manager 模板来部署 OpenShift Container Platform 集群所需的control plane 机器：

例 4.8. 05_control_plane.py Deployment Manager 模板

```
def GenerateConfig(context):

    resources = [{
        'name': context.properties['infra_id'] + '-master-0',
        'type': 'compute.v1.instance',
        'properties': {
            'disks': [{
                'autoDelete': True,
                'boot': True,
                'initializeParams': {
                    'diskSizeGb': context.properties['root_volume_size'],
                    'diskType': 'zones/' + context.properties['zones'][0] + '/diskTypes/pd-ssd',
                    'sourceImage': context.properties['image']
                }
            }],
            'machineType': 'zones/' + context.properties['zones'][0] + '/machineTypes/' +
context.properties['machine_type'],
            'metadata': {
                'items': [{
                    'key': 'user-data',
                    'value': context.properties['ignition']
                }]
            },
            'networkInterfaces': [{
                'subnetwork': context.properties['control_subnet']
            }],
            'serviceAccounts': [{
```

```
                'email': context.properties['service_account_email'],
                'scopes': ['https://www.googleapis.com/auth/cloud-platform']
            }],
            'tags': {
                'items': [
                    context.properties['infra_id'] + '-master',
                ]
            },
            'zone': context.properties['zones'][0]
        }
    }, {
        'name': context.properties['infra_id'] + '-master-1',
        'type': 'compute.v1.instance',
        'properties': {
            'disks': [{
                'autoDelete': True,
                'boot': True,
                'initializeParams': {
                    'diskSizeGb': context.properties['root_volume_size'],
                    'diskType': 'zones/' + context.properties['zones'][1] + '/diskTypes/pd-ssd',
                    'sourceImage': context.properties['image']
                }
            }],
            'machineType': 'zones/' + context.properties['zones'][1] + '/machineTypes/' +
context.properties['machine_type'],
            'metadata': {
                'items': [{
                    'key': 'user-data',
                    'value': context.properties['ignition']
                }]
            },
            'networkInterfaces': [{
                'subnetwork': context.properties['control_subnet']
            }],
            'serviceAccounts': [{
                'email': context.properties['service_account_email'],
                'scopes': ['https://www.googleapis.com/auth/cloud-platform']
            }],
            'tags': {
                'items': [
                    context.properties['infra_id'] + '-master',
                ]
            },
            'zone': context.properties['zones'][1]
        }
    }, {
        'name': context.properties['infra_id'] + '-master-2',
        'type': 'compute.v1.instance',
        'properties': {
            'disks': [{
                'autoDelete': True,
                'boot': True,
                'initializeParams': {
                    'diskSizeGb': context.properties['root_volume_size'],
                    'diskType': 'zones/' + context.properties['zones'][2] + '/diskTypes/pd-ssd',
                    'sourceImage': context.properties['image']
```

```
            }
        }],
        'machineType': 'zones/' + context.properties['zones'][2] + '/machineTypes/' +
context.properties['machine_type'],
        'metadata': {
            'items': [{
                'key': 'user-data',
                'value': context.properties['ignition']
            }]
        },
        'networkInterfaces': [{
            'subnetwork': context.properties['control_subnet']
        }],
        'serviceAccounts': [{
            'email': context.properties['service_account_email'],
            'scopes': ['https://www.googleapis.com/auth/cloud-platform']
        }],
        'tags': {
            'items': [
                context.properties['infra_id'] + '-master',
            ]
        },
        'zone': context.properties['zones'][2]
    }
}]

return {'resources': resources}
```

4.9.16. 等待 bootstrap 完成并删除 GCP 中的 bootstrap 资源

在 Google Cloud Platform (GCP) 中创建所有所需的基础架构后，请等待您通过安装程序生成的 Ignition 配置文件所置备的机器上完成 bootstrap 过程。

先决条件

- 配置 GCP 帐户。
- 为集群生成 Ignition 配置文件。
- 在 GCP 中创建和配置 VPC 及相关子网。
- 在 GCP 中创建和配置联网及负载均衡器。
- 创建 control plane 和计算角色。
- 创建 bootstrap 机器。
- 创建 control plane 机器。

流程

1. 更改到包含安装程序的目录，再运行以下命令：

```
$ ./openshift-install wait-for bootstrap-complete --dir <installation_directory> \ 1
    --log-level info 2
```

1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

2 要查看不同的安装详情，请指定 **warn**、**debug** 或 **error**，而不要指定 **info**。

如果命令退出且不显示 **FATAL** 警告，则代表您的 control plane 已被初始化。

2. 删除 bootstrap 资源：

```
$ gcloud compute backend-services remove-backend ${INFRA_ID}-api-internal-backend-
service --region=${REGION} --instance-group=${INFRA_ID}-bootstrap-instance-group --
instance-group-zone=${ZONE_0}
$ gsutil rm gs://${INFRA_ID}-bootstrap-ignition/bootstrap.ign
$ gsutil rb gs://${INFRA_ID}-bootstrap-ignition
$ gcloud deployment-manager deployments delete ${INFRA_ID}-bootstrap
```

4.9.17. 在 GCP 中创建额外的 worker 机器

您可以通过分散启动各个实例或利用集群外自动化流程（如自动缩放组），在 Google Cloud Platform (GCP) 中为您的集群创建 worker 机器。您还可以利用 OpenShift Container Platform 中的内置集群扩展机制和机器 API。

在本例中，您要使用 Deployment Manager 模板来手动启动一个实例。通过在文件中添加类型为 **06_worker.py** 的其他资源，即可启动其他实例。

注意

如果不使用提供的 Deployment Manager 模板来创建 worker 机器，您必须检查提供的信息并手动创建基础架构。如果集群没有正确初始化，您可能需要联系红帽支持并提供您的安装日志。

先决条件

- 配置 GCP 帐户。
- 为集群生成 Ignition 配置文件。
- 在 GCP 中创建和配置 VPC 及相关子网。
- 在 GCP 中创建和配置联网及负载均衡器。
- 创建 control plane 和计算角色。
- 创建 bootstrap 机器。
- 创建 control plane 机器。

流程

1. 复制本主题的 **worker 机器的 Deployment Manager 模板**部分中的模板，并将它以 **06_worker.py** 形式保存到计算机中。此模板描述了集群所需的 worker 机器。

2. 导出资源定义使用的变量。

   a. 导出托管计算机器的子网：

   ```
   $ export COMPUTE_SUBNET=(`gcloud compute networks subnets describe
   ${INFRA_ID}-worker-subnet --region=${REGION} --format json | jq -r .selfLink`)
   ```

   b. 为您的服务帐户导出电子邮件地址：

   ```
   $ export WORKER_SERVICE_ACCOUNT=(`gcloud iam service-accounts list --filter
   "email~^${INFRA_ID}-w@${PROJECT_NAME}." --format json | jq -r '.[0].email'`)
   ```

   c. 导出计算机器 Ignition 配置文件的位置：

   ```
   $ export WORKER_IGNITION=`cat <installation_directory>/worker.ign`
   ```

3. 创建 **06_worker.yaml** 资源定义文件：

```
$ cat <<EOF >06_worker.yaml
imports:
- path: 06_worker.py

resources:
- name: 'worker-0' (1)
  type: 06_worker.py
  properties:
    infra_id: '${INFRA_ID}' (2)
    zone: '${ZONE_0}' (3)
    compute_subnet: '${COMPUTE_SUBNET}' (4)
    image: '${CLUSTER_IMAGE}' (5)
    machine_type: 'n1-standard-4' (6)
    root_volume_size: '128'
    service_account_email: '${WORKER_SERVICE_ACCOUNT}' (7)
    ignition: '${WORKER_IGNITION}' (8)
- name: 'worker-1'
  type: 06_worker.py
  properties:
    infra_id: '${INFRA_ID}' (9)
    zone: '${ZONE_1}' (10)
    compute_subnet: '${COMPUTE_SUBNET}' (11)
    image: '${CLUSTER_IMAGE}' (12)
    machine_type: 'n1-standard-4' (13)
    root_volume_size: '128'
    service_account_email: '${WORKER_SERVICE_ACCOUNT}' (14)
    ignition: '${WORKER_IGNITION}' (15)
EOF
```

(1) **name** 是 worker 机器的名称，如 **worker-0**。

(2) (9) **infra_id** 是来自于提取步骤的 **INFRA_ID** 基础架构名称。

(3) (10) **zone** 是 worker 机器要部署到的区域，如 **us-central1-a**。

4 11 **compute_subnet** 是计算子网的 **selfLink**。

5 12 **image** 是 RHCOS 镜像的 **selfLink** URL。

6 13 **machine_type** 是实例的机器类型，如 **n1-standard-4**。

7 14 **service_account_email** 是创建的 worker 服务帐户的电子邮件地址。

8 15 **ignition** 是 **worker.ign** 文件的内容。

4. 可选：如果要启动更多实例，请在 **06_worker.yaml** 资源定义文件中包含类型为 **06_worker.py** 的其他资源。

5. 使用 **gcloud** CLI 创建部署：

```
$ gcloud deployment-manager deployments create ${INFRA_ID}-worker --config
06_worker.yaml
```

4.9.17.1. worker 机器的 Deloyment Manager 模板

您可以使用以下 Deployment Manager 模板来部署 OpenShift Container Platform 集群所需的 worker 机器：

例 4.9. 06_worker.py Deployment Manager 模板

```
def GenerateConfig(context):

    resources = [{
        'name': context.properties['infra_id'] + '-' + context.env['name'],
        'type': 'compute.v1.instance',
        'properties': {
            'disks': [{
                'autoDelete': True,
                'boot': True,
                'initializeParams': {
                    'diskSizeGb': context.properties['root_volume_size'],
                    'sourceImage': context.properties['image']
                }
            }],
            'machineType': 'zones/' + context.properties['zone'] + '/machineTypes/' +
context.properties['machine_type'],
            'metadata': {
                'items': [{
                    'key': 'user-data',
                    'value': context.properties['ignition']
                }]
            },
            'networkInterfaces': [{
                'subnetwork': context.properties['compute_subnet']
            }],
            'serviceAccounts': [{
                'email': context.properties['service_account_email'],
                'scopes': ['https://www.googleapis.com/auth/cloud-platform']
            }],
```

```
            'tags': {
                'items': [
                    context.properties['infra_id'] + '-worker',
                ]
            },
            'zone': context.properties['zone']
        }
    }]

    return {'resources': resources}
```

4.9.18. 通过下载二进制文件安装 OpenShift CLI

您需要安装 CLI（**oc**） 来使用命令行界面与 OpenShift Container Platform 进行交互。您可在 Linux 、Windows 或 macOS 上安装 **oc**。

重要

如果安装了旧版本的 **oc**，则无法使用 OpenShift Container Platform 4.6 中的所有命令。下载并安装新版本的 **oc**。

4.9.18.1. 在 Linux 上安装 OpenShift CLI

您可以按照以下流程在 Linux 上安装 OpenShift CLI（**oc**）二进制文件。

流程

1. 进入到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 在 **Version** 下拉菜单中选择相应的版本。
3. 单击 **OpenShift v4.6 Linux 客户端** 条目旁边的 **Download Now**，再保存文件。
4. 解包存档：

   ```
   $ tar xvzf <file>
   ```

5. 把 **oc** 二进制代码放到 **PATH** 中的目录中。
   执行以下命令可以查看当前的 **PATH** 设置：

   ```
   $ echo $PATH
   ```

安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
$ oc <command>
```

4.9.18.2. 在 Windows 上安装 OpenShift CLI

您可以按照以下流程在 Windows 上安装 OpenShift CLI（**oc**）二进制代码。

流程

1. 进入到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 在 **Version** 下拉菜单中选择相应的版本。
3. 单击 **OpenShift v4.6 Windows 客户端**条目旁边的 **Download Now**，再保存文件。
4. 使用 ZIP 程序解压存档。
5. 把 **oc** 二进制代码放到 **PATH** 中的目录中。
   要查看您的 **PATH**，请打开命令提示窗口并执行以下命令：

```
C:\> path
```

安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
C:\> oc <command>
```

4.9.18.3. 在 macOS 上安装 OpenShift CLI

您可以按照以下流程在 macOS 上安装 OpenShift CLI（**oc**）二进制代码。

流程

1. 进入到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 在 **Version** 下拉菜单中选择相应的版本。
3. 单击 **OpenShift v4.6 MacOSX 客户端**条目旁边的 **Download Now**，再保存文件。
4. 解包和解压存档。
5. 将 **oc** 二进制文件移到 PATH 的目录中。
   要查看您的 **PATH**，打开一个终端窗口并执行以下命令：

```
$ echo $PATH
```

安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
$ oc <command>
```

4.9.19. 使用 CLI 登录到集群

您可以通过导出集群 **kubeconfig** 文件，以默认系统用户身份登录集群。**kubeconfig** 文件包含关于集群的信息，供 CLI 用于将客户端连接到正确集群和 API 服务器。该文件特只适用于一个特定的集群，在 OpenShift Container Platform 安装过程中创建。

先决条件

- 已部署了 OpenShift Container Platform 集群。
- 已安装 **oc** CLI。

流程

1. 导出 **kubeadmin** 凭证：

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

2. 使用导出的配置，验证能否成功运行 **oc** 命令：

```
$ oc whoami
```

输出示例

```
system:admin
```

4.9.20. 批准机器的证书签名请求

将机器添加到集群时，会为您添加的每台机器生成两个待处理证书签名请求（CSR）。您必须确认这些 CSR 已获得批准，或根据需要自行批准。客户端请求必须首先被批准，然后是服务器请求。

先决条件

- 您已将机器添加到集群中。

流程

1. 确认集群可以识别这些机器：

```
$ oc get nodes
```

输出示例

```
NAME     STATUS   ROLES   AGE  VERSION
master-0 Ready    master  63m  v1.19.0
master-1 Ready    master  63m  v1.19.0
master-2 Ready    master  64m  v1.19.0
```

输出将列出您创建的所有机器。

注意

在一些 CSR 被批准前，以上输出可能不包括计算节点（也称为 worker 节点）。

2. 检查待处理的 CSR，并确保可以看到添加到集群中的每台机器都有 **Pending** 或 **Approved** 状态的客户端请求：

```
$ oc get csr
```

输出示例

```
NAME       AGE     REQUESTOR                                    CONDITION
```

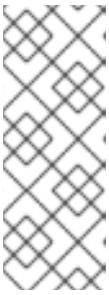
```
csr-8b2br   15m    system:serviceaccount:openshift-machine-config-operator:node-bootstrapper   Pending
csr-8vnps   15m    system:serviceaccount:openshift-machine-config-operator:node-bootstrapper   Pending
...
```

在本例中，两台机器加入了集群。您可能会在列表中看到更多已批准的 CSR。

3. 如果 CSR 没有获得批准，请在所添加机器的所有待处理 CSR 都处于 **Pending** 状态后，为您的集群机器批准这些 CSR：

注意

由于 CSR 会自动轮转，因此请在将机器添加到集群后一小时内批准您的 CSR。如果没有在一小时内批准，证书将会轮转，每个节点将会存在多个证书。您必须批准所有这些证书。批准客户端 CSR 后， Kubelet 为服务证书创建辅助 CSR，这需要手动批准。然后，如果 Kubelet 请求具有相同参数的新证书，则 **machine-approver** 会自动批准后续服务证书续订请求。



注意

对于在未启用机器 API 的平台中运行的集群，如裸机和其他用户置备的基础架构，必须采用一种方法自动批准 kubelet 提供证书请求（CSR）。如果没有批准请求，则 **oc exec**、**oc rsh** 和 **oc logs** 命令将无法成功，因为 API 服务器连接到 kubelet 时需要服务证书。与 Kubelet 端点联系的任何操作都需要此证书批准。这个方法必须监视新的 CSR，确认 CSR 由 **system:node** 或 **system:admin** 组中的 **node-bootstrapper** 服务帐户提交，并确认节点的身份。

- 若要单独批准，请对每个有效的 CSR 运行以下命令：

```
$ oc adm certificate approve <csr_name> 1
```

1 **<csr_name>** 是当前 CSR 列表中 CSR 的名称。

- 要批准所有待处理的 CSR，请运行以下命令：

```
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}{{"\n"}}{{end}}{{end}}' | xargs --no-run-if-empty oc adm certificate approve
```

注意

在有些 CSR 被批准前，一些 Operator 可能无法使用。

4. 现在，您的客户端请求已被批准，您必须查看添加到集群中的每台机器的服务器请求：

```
$ oc get csr
```

输出示例

```
NAME        AGE     REQUESTOR                                                   CONDITION
csr-bfd72   5m26s   system:node:ip-10-0-50-126.us-east-2.compute.internal
```

```
Pending
csr-c57lv   5m26s   system:node:ip-10-0-95-157.us-east-2.compute.internal
Pending
...
```

5. 如果剩余的 CSR 没有被批准，且处于 **Pending** 状态，请批准集群机器的 CSR：

- 若要单独批准，请对每个有效的 CSR 运行以下命令：

```
$ oc adm certificate approve <csr_name> 1
```

1 **<csr_name>** 是当前 CSR 列表中 CSR 的名称。

- 要批准所有待处理的 CSR，请运行以下命令：

```
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}{{"\n"}}
{{end}}{{end}}' | xargs oc adm certificate approve
```

6. 批准所有客户端和服务器 CSR 后，器将处于 **Ready** 状态。运行以下命令验证：

```
$ oc get nodes
```

输出示例

```
NAME      STATUS    ROLES   AGE  VERSION
master-0  Ready     master  73m  v1.20.0
master-1  Ready     master  73m  v1.20.0
master-2  Ready     master  74m  v1.20.0
worker-0  Ready     worker  11m  v1.20.0
worker-1  Ready     worker  11m  v1.20.0
```

注意

批准服务器 CSR 后可能需要几分钟时间让机器转换为 **Ready** 状态。

其他信息

- 如需有关 CSR 的更多信息，请参阅证书签名请求。

4.9.21. 可选：添加入口 DNS 记录

如果您在创建 Kubernetes 清单并生成 Ignition 配置时删除了 DNS 区配置，您必须手动创建指向入口负载均衡器的 DNS 记录。您可以创建通配符 ***.apps.{baseDomain}.** 或具体的记录。您可以根据自己的要求使用 A、CNAME 和其他记录。

先决条件

- 配置 GCP 帐户。
- 在创建 Kubernetes 清单并生成 Ignition 配置时删除 DNS 区配置。
- 在 GCP 中创建和配置 VPC 及相关子网。

- 在 GCP 中创建和配置联网及负载均衡器。
- 创建 control plane 和计算角色。
- 创建 bootstrap 机器。
- 创建 control plane 机器。
- 创建 worker 机器。

流程

1. 等待入口路由器创建负载均衡器并填充 **EXTERNAL-IP** 字段：

```
$ oc -n openshift-ingress get service router-default
```

输出示例

```
NAME            TYPE           CLUSTER-IP     EXTERNAL-IP     PORT(S)                      AGE
router-default  LoadBalancer  172.30.18.154  35.233.157.184
80:32288/TCP,443:31215/TCP  98
```

2. 在您的区中添加 A 记录：

- 使用 A 记录：

i. 为路由器 IP 地址导出变量：

```
$ export ROUTER_IP=`oc -n openshift-ingress get service router-default --no-
headers | awk '{print $4}'`
```

ii. 在专用区中添加 A 记录：

```
$ if [ -f transaction.yaml ]; then rm transaction.yaml; fi
$ gcloud dns record-sets transaction start --zone ${INFRA_ID}-private-zone
$ gcloud dns record-sets transaction add ${ROUTER_IP} --name
\*.apps.${CLUSTER_NAME}.${BASE_DOMAIN}. --ttl 300 --type A --zone
${INFRA_ID}-private-zone
$ gcloud dns record-sets transaction execute --zone ${INFRA_ID}-private-zone
```

iii. 对于外部集群，还要将 A 记录添加到公共区：

```
$ if [ -f transaction.yaml ]; then rm transaction.yaml; fi
$ gcloud dns record-sets transaction start --zone ${BASE_DOMAIN_ZONE_NAME}
$ gcloud dns record-sets transaction add ${ROUTER_IP} --name
\*.apps.${CLUSTER_NAME}.${BASE_DOMAIN}. --ttl 300 --type A --zone
${BASE_DOMAIN_ZONE_NAME}
$ gcloud dns record-sets transaction execute --zone
${BASE_DOMAIN_ZONE_NAME}
```

- 如果需要添加特定域而不使用通配符，可以为集群的每个当前路由创建条目：

```
$ oc get --all-namespaces -o jsonpath='{range .items[*]}{range .status.ingress[*]}{.host}
{"\n"}{end}{end}' routes
```

输出示例

```
oauth-openshift.apps.your.cluster.domain.example.com
console-openshift-console.apps.your.cluster.domain.example.com
downloads-openshift-console.apps.your.cluster.domain.example.com
alertmanager-main-openshift-monitoring.apps.your.cluster.domain.example.com
grafana-openshift-monitoring.apps.your.cluster.domain.example.com
prometheus-k8s-openshift-monitoring.apps.your.cluster.domain.example.com
```

4.9.22. 在用户置备的基础架构上完成 GCP 安装

在 Google Cloud Platform (GCP) 用户置备的基础架构上启动 OpenShift Container Platform 安装后，您可以监控集群事件，直到集群就绪可用。

先决条件

- 在用户置备的 GCP 基础架构上为 OpenShift Container Platform 集群部署 bootstrap 机器。
- 安装 **oc** CLI 并登录。

流程

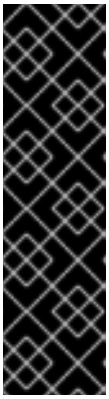
1. 完成集群安装：

```
$ ./openshift-install --dir <installation_directory> wait-for install-complete 1
```

输出示例

```
INFO Waiting up to 30m0s for the cluster to initialize...
```

1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。



重要

- 安装程序生成的 Ignition 配置文件包含在 24 小时后过期的证书，然后在过期时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外情况是，您需要手动批准待处理的 **node-bootstrapper** 证书签名请求（CSR）来恢复 kubelet 证书。如需更多信息，请参阅*从过期的 control plane 证书中恢复*的文档。
- 建议您在生成 12 小时后使用 Ignition 配置文件，因为集群安装后 24 小时证书从 16 小时轮转至 22 小时。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中运行证书更新时避免安装失败。

2. 观察集群的运行状态。

   a. 运行以下命令来查看当前的集群版本和状态：

```
$ oc get clusterversion
```

输出示例

```
-
```

```
NAME     VERSION   AVAILABLE   PROGRESSING   SINCE   STATUS
version            False       True          24m     Working towards 4.5.4: 99% complete
```

b. 运行以下命令，查看 control plane 上由 Cluster Version Operator (CVO) 管理的 Operator：

```
$ oc get clusteroperators
```

输出示例

```
NAME                                       VERSION   AVAILABLE   PROGRESSING   DEGRADED
SINCE
authentication                             4.5.4     True        False         False      7m56s
cloud-credential                           4.5.4     True        False         False      31m
cluster-autoscaler                         4.5.4     True        False         False      16m
console                                    4.5.4     True        False         False      10m
csi-snapshot-controller                    4.5.4     True        False         False      16m
dns                                        4.5.4     True        False         False      22m
etcd                                       4.5.4     False       False         False      25s
image-registry                             4.5.4     True        False         False      16m
ingress                                    4.5.4     True        False         False      16m
insights                                   4.5.4     True        False         False      17m
kube-apiserver                             4.5.4     True        False         False      19m
kube-controller-manager                    4.5.4     True        False         False      20m
kube-scheduler                             4.5.4     True        False         False      20m
kube-storage-version-migrator              4.5.4     True        False         False      16m
machine-api                                4.5.4     True        False         False      22m
machine-config                             4.5.4     True        False         False      22m
marketplace                                4.5.4     True        False         False      16m
monitoring                                 4.5.4     True        False         False      10m
network                                    4.5.4     True        False         False      23m
node-tuning                                4.5.4     True        False         False      23m
openshift-apiserver                        4.5.4     True        False         False      17m
openshift-controller-manager               4.5.4     True        False         False      15m
openshift-samples                          4.5.4     True        False         False      16m
operator-lifecycle-manager                 4.5.4     True        False         False      22m
operator-lifecycle-manager-catalog         4.5.4     True        False         False      22m
operator-lifecycle-manager-packageserver   4.5.4     True        False         False      18m
service-ca                                 4.5.4     True        False         False      23m
service-catalog-apiserver                  4.5.4     True        False         False      23m
service-catalog-controller-manager         4.5.4     True        False         False      23m
storage                                    4.5.4     True        False         False      17m
```

c. 运行以下命令来查看您的集群 pod：

```
$ oc get pods --all-namespaces
```

输出示例

```
NAMESPACE                                       NAME
READY   STATUS    RESTARTS   AGE
kube-system                                     etcd-member-ip-10-0-3-111.us-east-
2.compute.internal          1/1     Running   0          35m
kube-system                                     etcd-member-ip-10-0-3-239.us-east-
2.compute.internal          1/1     Running   0          37m
```

```
kube-system                                        etcd-member-ip-10-0-3-24.us-east-
2.compute.internal              1/1     Running     0          35m
openshift-apiserver-operator                       openshift-apiserver-operator-6d6674f4f4-
h7t2t                           1/1     Running     1          37m
openshift-apiserver                                apiserver-fm48r
1/1     Running     0          30m
openshift-apiserver                                apiserver-fxkvv
1/1     Running     0          29m
openshift-apiserver                                apiserver-q85nm
1/1     Running     0          29m
...
openshift-service-ca-operator                      openshift-service-ca-operator-66ff6dc6cd-
9r257                           1/1     Running     0          37m
openshift-service-ca                               apiservice-cabundle-injector-695b6bcbc-cl5hm
1/1     Running     0          35m
openshift-service-ca                               configmap-cabundle-injector-8498544d7-
25qn6                           1/1     Running     0          35m
openshift-service-ca                               service-serving-cert-signer-6445fc9c6-wqdqn
1/1     Running     0          35m
openshift-service-catalog-apiserver-operator       openshift-service-catalog-apiserver-
operator-549f44668b-b5q2w       1/1     Running     0          32m
openshift-service-catalog-controller-manager-operator   openshift-service-catalog-
controller-manager-operator-b78cr2lnm   1/1     Running     0          31m
```

当前集群版本是 **AVAILABLE** 时表示安装完毕。

4.9.23. OpenShift Container Platform 的 Telemetry 访问

在 OpenShift Container Platform 4.6 中，默认运行的 Telemetry 服务提供有关集群健康状况和成功更新的指标，需要访问互联网。如果您的集群连接到互联网，Telemetry 会自动运行，而且集群会注册到 OpenShift Cluster Manager。

确认 OpenShift Cluster Manager 清单正确后，可以由 Telemetry 自动维护，也可以使用 OpenShift Cluster Manager 手动维护，使用订阅监控来跟踪帐户或多集群级别的 OpenShift Container Platform 订阅。

其他资源

- 有关 Telemetry 服务的更多信息，请参阅关于远程健康监控。

4.9.24. 后续步骤

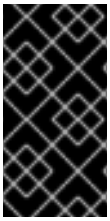
- 自定义集群。
- 如果需要，您可以选择不使用远程健康报告。

4.10. 使用 DEPLOYMENT MANAGER 模板在 GCP 上将集群安装到共享 VPC 中

在 OpenShift Container Platform 版本 4.6 中，您可以使用您提供的基础架构在 Google Cloud Platform（GCP） 上安装共享 VPC 的集群。在这种情况下，安装到共享 VPC 的集群是一个集群，被配置为使用与部署集群不同的项目中的 VPC。

共享 VPC 可让机构将资源从多个项目连接到通用 VPC 网络。您可以使用该网络的内部 IP 来安全有效地在机构内进行通信。如需有关共享 VPC 的更多信息，请参阅 GCP 文档中的共享 VPC 概述。

此处概述了进行用户提供基础架构安装到共享 VPC 的步骤。提供的几个 Deployment Manager 模板可协助完成这些步骤，也可帮助您自行建模。您还可以自由选择通过其他方法创建所需的资源。



重要

进行用户置备的基础架构安装的步骤仅作为示例。使用您提供的基础架构安装集群需要了解云供应商和 OpenShift Container Platform 安装过程。提供的几个 Deployment Manager 模板可帮助完成这些步骤，或者帮助您自行建模。您也可以自由选择通过其他方法创建所需的资源；模板仅作示例之用。

4.10.1. 先决条件

- 查看有关 OpenShift Container Platform 安装和更新流程的详细信息。
- 如果使用防火墙并计划使用遥测（telemetry），您必须将防火墙配置为允许集群需要访问的站点。
- 如果不允许系统管理身份和访问管理（IAM），集群管理员可以 手动创建和维护 IAM 凭证。手动模式也可以用于云 IAM API 无法访问的环境中。

注意

如果您要配置代理，请务必也要查看此站点列表。

4.10.2. 证书签名请求管理

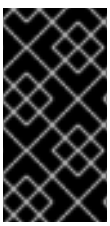
在使用您置备的基础架构时，集群只能有限地访问自动机器管理，因此您必须提供一种在安装后批准集群证书签名请求 (CSR) 的机制。**kube-controller-manager** 只能批准 kubelet 客户端 CSR。 **machine-approver** 无法保证使用 kubelet 凭证请求的提供证书的有效性，因为它不能确认是正确的机器发出了该请求。您必须决定并实施一种方法，以验证 kubelet 提供证书请求的有效性并进行批准。

4.10.3. OpenShift Container Platform 的互联网访问

在 OpenShift Container Platform 4.6 中，您需要访问互联网来安装集群。

您必须具有以下互联网访问权限：

- 访问 OpenShift Cluster Manager 以下载安装程序并执行订阅管理。如果集群可以访问互联网，并且没有禁用 Telemetry，该服务会自动授权您的集群。
- 访问 Quay.io，以获取安装集群所需的软件包。
- 获取执行集群更新所需的软件包。



重要

如果您的集群无法直接访问互联网，则可以在置备的某些类基础架构上执行受限网络安装。在此过程中，您要下载所需的内容，并使用它在镜像 registry（mirror registry） 中填充安装集群并生成安装程序所需的软件包。对于某些安装类型，集群要安装到的环境不需要访问互联网。在更新集群之前，要更新 registry 镜像系统中的内容。

4.10.4. 配置托管集群的 GCP 项目

在安装 OpenShift Container Platform 之前，您必须配置 Google Cloud Platform (GCP) 项目来托管它。

4.10.4.1. 创建一个 GCP 项目

为了安装 OpenShift Container Platform，您必须在您的 Google Cloud Platform (GCP) 账户中创建一个项目来托管它。

流程

- 创建一个项目来托管您的 OpenShift Container Platform 集群。请参阅 GCP 文档中的创建和管理项目。

重要

如果您使用安装程序置备的基础架构，您的 GCP 项目必须使用 Premium Network Service Tier。使用安装程序安装的集群不支持 Standard Network Service Tier。安装程序为 **api-int.<cluster_name>.<base_domain>** URL 配置内部负载均衡；内部负载均衡需要 Premium Tier。

4.10.4.2. 在 GCP 中启用 API 服务

Google Cloud Platform (GCP) 项目需要访问一些 API 服务来完成 OpenShift Container Platform 安装。

先决条件

- 创建了用于托管集群的项目。

流程

- 在托管集群的项目中启用如下所需的 API 服务。请参阅 GCP 文档中的启用服务。

表 4.37. 所需的 API 服务

API 服务	控制台服务名称
Cloud Deployment Manager V2 API	**deploymentmanager.googleapis.com**
compute Engine API	**compute.googleapis.com**
Google Cloud API	**cloudapis.googleapis.com**
Cloud Resource Manager API	**cloudresourcemanager.googleapis.com**
Google DNS API	**dns.googleapis.com**
IAM Service Account Credentials API	**iamcredentials.googleapis.com**
Identity and Access Management (IAM) API	**iam.googleapis.com**

API 服务	控制台服务名称
Service Management API	**servicemanagement.googleapis.com**
Service Usage API	**serviceusage.googleapis.com**
Google Cloud Storage JSON API	**storage-api.googleapis.com**
Cloud Storage	**storage-component.googleapis.com**

4.10.4.3. GCP 帐户限值

OpenShift Container Platform 集群使用诸多 Google Cloud Platform (GCP) 组件，默认的配额不会影响您安装默认 OpenShift Container Platform 集群的能力。

默认集群中包含三台计算机器和三台 control plane 机器，使用了以下资源。请注意，一些资源只在 bootstrap 过程中需要，会在集群部署后删除。

表 4.38. 默认集群中使用的 GCP 资源

服务	组件	位置	所需的资源总数	bootstrap 后删除的资源
服务帐户	IAM	全局	5	0
防火墙规则	网络	全局	11	1
转发规则	Compute	全局	2	0
健康检查	Compute	全局	2	0
镜像	Compute	全局	1	0
网络	网络	全局	1	0
路由器	网络	全局	1	0
Routes	网络	全局	2	0
子网	Compute	全局	2	0
目标池	网络	全局	2	0

注意

如果在安装过程中存在任何配额不足的情况，安装程序会显示一个错误信息，包括超过哪个配额，以及相关的区域。

请考虑您的集群的实际大小、预定的集群增长以及来自与您的帐户关联的其它集群的使用情况。CPU、静态 IP 地址和持久性磁盘 SSD (storage) 配额是最可能不足的。

如果您计划在以下区域之一部署集群，您将超过最大存储配额，并可能会超过 CPU 配额限制：

- **asia-east2**
- **asia-northeast2**
- **asia-south1**
- **domain-southeast1**
- **europe-north1**
- **europe-west2**
- **europe-west3**
- **europe-west6**
- **northamerica-northeast1**
- **southamerica-east1**
- **us-west2**

您可以从 GCP 控制台增加资源配额，但可能需要提交一个支持问题单。务必提前规划集群大小，以便在安装 OpenShift Container Platform 集群前有足够的时间来等待支持问题单被处理。

4.10.4.4. 在 GCP 中创建服务帐户

OpenShift Container Platform 需要一个 Google Cloud Platform (GCP) 服务帐户，用于访问 Google API 中数据的验证和授权。如果您没有包含项目中所需角色的现有 IAM 服务帐户，您必须创建一个。

先决条件

- 创建了用于托管集群的项目。

流程

1. 在用于托管 OpenShift Container Platform 集群的项目中，创建一个新服务帐户。请参阅 GCP 文档中的创建服务账户。
2. 为服务帐户授予适当的权限。您可以逐一授予权限，也可以为其分配 **Owner** 角色。请参阅将角色授予特定资源的服务帐户。

注意

获得所需权限最简单的方法是把服务帐户设置为项目的拥有者（owner），这意味着该服务帐户对项目有完全的控制权。您必须考虑这样设定所带来的的风险是否可以接受。

3. 以 JSON 格式创建服务帐户密钥。请参阅 GCP 文档中的创建服务帐户密钥。
创建集群时需要该服务帐户密钥。

4.10.4.4.1. 所需的 GCP 权限

如果将 **Owner** 角色附加到您创建的服务帐户，您向该服务帐户授予所有的权限，包括安装 OpenShift Container Platform 所需的权限。要部署 OpenShift Container Platform 集群，服务帐户需要以下权限：如果您将集群部署到现有的 VPC 中，则服务帐户不需要某些网络权限，如下表所示：

安装程序所需的角色

- Compute Admin
- Security Admin
- Service Account Admin
- Service Account User
- Storage Admin

安装过程中创建网络资源所需的角色

- DNS Administrator

用户置备的 GCP 基础架构所需的角色

- Deployment Manager Editor
- Service Account Key Admin

可选角色

若要使集群为其 Operator 创建新的有限凭证，请添加以下角色：

- Service Account Key Admin

以下角色将应用到 control plane 或计算机器使用的服务帐户：

表 4.39. GCP 服务帐户权限

帐户	角色
Control Plane	**roles/compute.instanceAdmin**
	roles/compute.networkAdmin
	roles/compute.securityAdmin
	roles/storage.admin
	roles/iam.serviceAccountUser
Compute	**roles/compute.viewer**
	roles/storage.admin

4.10.4.5. 支持的 GCP 区域

您可以将 OpenShift Container Platform 集群部署到下列 Google Cloud Platform (GCP) 区域：

- **asia-east1** (Changhua County, Taiwan)
- **asia-east2** (Hong Kong)
- **asia-northeast1** (Tokyo, Japan)
- **asia-northeast2** (Osaka, Japan)
- **asia-northeast3** (Seoul, South Korea)
- **asia-south1** (Mumbai, India)
- **asia-southeast1** (Jurong West, Singapore)
- **asia-southeast2** (Jakarta, India)
- **australia-southeast1** (Sydney, Australia)
- **europe-north1** (Hamina, Finland)
- **europe-west1** (St. Ghislain, Belgium)
- **europe-west2** (London, England, UK)
- **europe-west3** (Frankfurt, Germany)
- **europe-west4** (Eemshaven, Netherlands)
- **europe-west6** (Zürich, Switzerland)
- **northamerica-northeast1** (Montréal, Québec, Canada)
- **southamerica-east1** (São Paulo, Brazil)
- **us-central1** (Council Bluffs, Iowa, USA)
- **us-east1** (Moncks Corner, South Carolina, USA)
- **us-east4** (Ashburn, Northern Virginia, USA)
- **us-west1** (The Dalles, Oregon, USA)
- **us-west2** (Los Angeles, California, USA)
- **us-west3** (Salt Lake City, Utah, USA)
- **us-west4** (Las Vegas, Nevada, USA)

4.10.4.6. 为 GCP 安装和配置 CLI 工具

要使用用户置备的基础架构在 Google Cloud Platform (GCP) 上安装 OpenShift Container Platform，您必须为 GCP 安装和配置 CLI 工具。

先决条件

- 创建了用于托管集群的项目。
- 您创建了服务帐户，并授予其所需的权限。

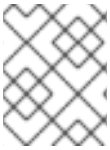
流程

1. 在 **$PATH** 中安装以下二进制文件：
   - **gcloud**
   - **gsutil**

   请参阅 GCP 文档中的安装最新 Cloud SDK 版本。
2. 使用 **gcloud** 工具及您配置的服务帐户进行身份验证。
   请参阅 GCP 文档中的使用服务帐户授权。

4.10.5. 配置托管共享 VPC 网络的 GCP 项目

如果使用共享的 Virtual Private Cloud（VPC）在 Google Cloud Platform（GCP）中托管 OpenShift Container Platform 集群，您必须配置托管它的项目。

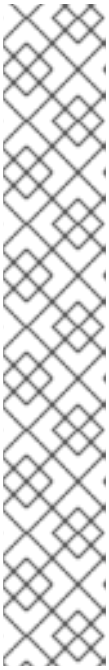


注意

如果您已有托管共享 VPC 网络的项目，请查看本节以确保项目满足安装 OpenShift Container Platform 集群的所有要求。

流程

1. 创建一个项目来托管您的 OpenShift Container Platform 集群。请参阅 GCP 文档中的创建和管理项目。
2. 在托管共享 VPC 的项目中创建服务帐户。请参阅 GCP 文档中的创建服务账户。
3. 为服务帐户授予适当的权限。您可以逐一授予权限，也可以为其分配 **Owner** 角色。请参阅将角色授予特定资源的服务帐户。



注意

获得所需权限最简单的方法是把服务帐户设置为项目的拥有者（owner），这意味着该服务帐户对项目有完全的控制权。您必须考虑这样设定所带来的的风险是否可以接受。

托管共享 VPC 网络的项目的服务帐户需要以下角色：

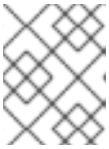
- Compute 网络用户
- Compute Security Admin
- Deployment Manager Editor
- DNS Administrator
- Security Admin
- 网络管理管理员

4.10.5.1. 为 GCP 配置 DNS

要安装 OpenShift Container Platform，您使用的 Google Cloud Platform (GCP) 帐户必须在项目中有一个专门的公共托管的区，它托管您的集群要安装到的共享VPC。此区域必须对域具有权威。DNS 服务为集群外部连接提供集群 DNS 解析和名称查询。

流程

1. 标识您的域或子域，以及注册商（registrar）。您可以转移现有的域和注册商，或通过 GCP 或其他来源获取新的域和注册商。



注意

如果您购买新域，则需要时间来传播相关的 DNS 更改。有关通过 Google 购买域的更多信息，请参阅 Google Domains。

2. 在 GCP 项目中为您的域或子域创建一个公共托管区。请参阅 GCP 文档中的创建公共区。使用合适的根域（如 **openshiftcorp.com**）或子域（如 **clusters.openshiftcorp.com**）。
3. 从托管区记录中提取新的权威名称服务器。请参阅 GCP 文档中的查找您的云 DNS 名称服务器。您通常有四个名称服务器。
4. 更新域所用名称服务器的注册商记录。例如，如果您将域注册到了 Google Domains，请参阅 Google Domains 帮助中的以下主题：如何切换到自定义名称服务器。
5. 如果您将根域迁移到 Google Cloud DNS，请迁移您的 DNS 记录。请参阅 GCP 文档中的 Migrating to Cloud DNS 。
6. 如果您使用子域，请按照您公司的流程将其委派记录添加到父域。这个过程可能包括对您公司的 IT 部门或控制您公司的根域和 DNS 服务的部门提出请求。

4.10.5.2. 在 GCP 中创建 VPC

您必须在 Google Cloud Platform (GCP) 中创建一个 VPC，供您的 OpenShift Container Platform 集群使用。您可以自定义 VPC 来满足您的要求。创建 VPC 的一种方法是修改提供的 Deployment Manager 模板。

注意

如果不使用提供的 Deployment Manager 模板来创建 GCP 基础架构，您必须检查提供的信息并手动创建基础架构。如果集群没有正确初始化，您可能需要联系红帽支持并提供您的安装日志。

先决条件

- 配置 GCP 帐户。

流程

1. 复制 **VPC 的 Deployment Manager 模板**一节中的模板，并将它以 **01_vpc.py** 形式保存到计算机上。此模板描述了集群所需的 VPC。
2. 导出资源定义所需的以下变量：
   a. 导出 control plane CIDR:

```
$ export MASTER_SUBNET_CIDR='10.0.0.0/19'
```

   b. 导出计算 CIDR:

```
$ export WORKER_SUBNET_CIDR='10.0.32.0/19'
```

   c. 将部署 VPC 网络和集群的区域导出到：

```
$ export REGION='<region>'
```

3. 导出托管共享 VPC 的项目 ID 的变量：

```
$ export HOST_PROJECT=<host_project>
```

4. 导出属于主机项目的服务帐户电子邮件的变量：

```
$ export HOST_PROJECT_ACCOUNT=<host_service_account_email>
```

5. 创建 **01_vpc.yaml** 资源定义文件：

```
$ cat <<EOF >01_vpc.yaml
imports:
- path: 01_vpc.py

resources:
- name: cluster-vpc
  type: 01_vpc.py
  properties:
    infra_id: '<prefix>' 1
    region: '${REGION}' 2
```

```
    master_subnet_cidr: '${MASTER_SUBNET_CIDR}' 3
    worker_subnet_cidr: '${WORKER_SUBNET_CIDR}' 4
EOF
```

1. **infra_id** 是网络名称的前缀。
2. **region** 是集群要部署到的区域，如 **us-central1**。
3. **master_subnet_cidr** 是 master 子网的 CIDR，如 **10.0.0.0/19**。
4. **worker_subnet_cidr** 是 worker 子网的 CIDR，如 **10.0.32.0/19**。

6. 使用 **gcloud** CLI 创建部署：

```
$ gcloud deployment-manager deployments create <vpc_deployment_name> --config
01_vpc.yaml --project ${HOST_PROJECT} --account ${HOST_PROJECT_ACCOUNT} 1
```

1. 对于 **<vpc_deployment_name>**，请指定要部署的 VPC 名称。

7. 导出其他组件需要的 VPC 变量：
   a. 导出主机项目网络的名称：

```
$ export HOST_PROJECT_NETWORK=<vpc_network>
```

   b. 导出主机项目 control plane 子网的名称：

```
$ export HOST_PROJECT_CONTROL_SUBNET=<control_plane_subnet>
```

   c. 导出主机项目计算子网的名称：

```
$ export HOST_PROJECT_COMPUTE_SUBNET=<compute_subnet>
```

8. 设置共享 VPC。请参阅 GCP 文档中的 设置共享 VPC。

4.10.5.2.1. VPC 的 Deployment Manager 模板

您可以使用以下 Deployment Manager 模板来部署 OpenShift Container Platform 集群所需的 VPC：

例 4.10. 01_VPC.py Deployment Manager 模板

```
def GenerateConfig(context):

  resources = [{
    'name': context.properties['infra_id'] + '-network',
    'type': 'compute.v1.network',
    'properties': {
        'region': context.properties['region'],
        'autoCreateSubnetworks': False
    }
  }, {
    'name': context.properties['infra_id'] + '-master-subnet',
```

```
        'type': 'compute.v1.subnetwork',
        'properties': {
            'region': context.properties['region'],
            'network': '$(ref.' + context.properties['infra_id'] + '-network.selfLink)',
            'ipCidrRange': context.properties['master_subnet_cidr']
        }
    }, {
        'name': context.properties['infra_id'] + '-worker-subnet',
        'type': 'compute.v1.subnetwork',
        'properties': {
            'region': context.properties['region'],
            'network': '$(ref.' + context.properties['infra_id'] + '-network.selfLink)',
            'ipCidrRange': context.properties['worker_subnet_cidr']
        }
    }, {
        'name': context.properties['infra_id'] + '-router',
        'type': 'compute.v1.router',
        'properties': {
            'region': context.properties['region'],
            'network': '$(ref.' + context.properties['infra_id'] + '-network.selfLink)',
            'nats': [{
                'name': context.properties['infra_id'] + '-nat-master',
                'natIpAllocateOption': 'AUTO_ONLY',
                'minPortsPerVm': 7168,
                'sourceSubnetworkIpRangesToNat': 'LIST_OF_SUBNETWORKS',
                'subnetworks': [{
                    'name': '$(ref.' + context.properties['infra_id'] + '-master-subnet.selfLink)',
                    'sourceIpRangesToNat': ['ALL_IP_RANGES']
                }]
            }, {
                'name': context.properties['infra_id'] + '-nat-worker',
                'natIpAllocateOption': 'AUTO_ONLY',
                'minPortsPerVm': 512,
                'sourceSubnetworkIpRangesToNat': 'LIST_OF_SUBNETWORKS',
                'subnetworks': [{
                    'name': '$(ref.' + context.properties['infra_id'] + '-worker-subnet.selfLink)',
                    'sourceIpRangesToNat': ['ALL_IP_RANGES']
                }]
            }]
        }
    }]

    return {'resources': resources}
```

4.10.6. 为 GCP 创建安装文件

要使用用户置备的基础架构在 Google Cloud Platform (GCP) 上安装 OpenShift Container Platform，您必须生成并修改安装程序部署集群所需的文件，以便集群只创建要使用的机器。您要生成并自定义 **install-config.yaml** 文件、Kubernetes 清单和 Ignition 配置文件。您也可以选择在安装准备阶段首先设置独立的 **var** 分区。

4.10.6.1. 手动创建安装配置文件

对于使用用户置备的基础架构的 OpenShift Container Platform 安装，您必须手动生成安装配置文件。

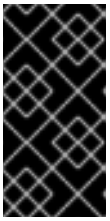
先决条件

- 获取 OpenShift Container Platform 安装程序和集群的访问令牌。

流程

1. 创建用来存储您所需的安装资产的安装目录：

```
$ mkdir <installation_directory>
```



重要

您必须创建目录。一些安装信息，如 bootstrap X.509 证书，有较短的过期间隔，因此不要重复使用安装目录。如果要重复使用另一个集群安装中的个别文件，可以将其复制到您的目录中。但是，一些安装数据的文件名可能会在发行版本之间有所改变。从 OpenShift Container Platform 老版本中复制安装文件时要格外小心。

2. 自定义以下 **install-config.yaml** 文件模板，并将它保存到 **<installation_directory>** 中。

注意

此配置文件必须命名为 **install-config.yaml**。

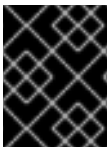
3. 备份 **install-config.yaml** 文件，以便用于安装多个集群。

重要

install-config.yaml 文件会在安装过程的下一步骤中消耗掉。现在必须备份它。

4.10.6.2. GCP 自定义 install-config.yaml 文件示例

您可以自定义 **install-config.yaml** 文件，以指定有关 OpenShift Container Platform 集群平台的更多信息，或修改所需参数的值。



重要

此示例 YAML 文件仅供参考。您必须使用安装程序来获取 **install-config.yaml** 文件，并且修改该文件。

```
apiVersion: v1
baseDomain: example.com (1)
controlPlane: (2)
  hyperthreading: Enabled (3) (4)
  name: master
  platform:
    gcp:
      type: n2-standard-4
      zones:
      - us-central1-a
      - us-central1-c
  replicas: 3
compute: (5)
```

```
- hyperthreading: Enabled 6
  name: worker
  platform:
    gcp:
      type: n2-standard-4
      zones:
      - us-central1-a
      - us-central1-c
  replicas: 0
metadata:
  name: test-cluster
networking:
  clusterNetwork:
  - cidr: 10.128.0.0/14
    hostPrefix: 23
  machineNetwork:
  - cidr: 10.0.0.0/16
  networkType: OpenShiftSDN
  serviceNetwork:
  - 172.30.0.0/16
platform:
  gcp:
    projectID: openshift-production 7
    region: us-central1 8
pullSecret: '{"auths": ...}'
fips: false 9
sshKey: ssh-ed25519 AAAA... 10
publish: Internal 11
```

1 指定主机项目上的公共 DNS。

2 5 如果没有提供这些参数和值，安装程序会提供默认值。

3 6 **controlPlane** 部分是一个单映射，但 compute 部分是一系列映射。为满足不同数据结构的要求，**compute** 部分的第一行必须以连字符 **-** 开头，**controlPlane** 部分的第一行则不可以连字符开头。虽然这两个部分目前都定义单个机器池，但未来的 OpenShift Container Platform 版本可能会支持在安装过程中定义多个计算池。只使用一个 control plane 池。

4 是否要启用或禁用并发多线程或**超线程**。默认情况下，启用并发多线程以提高机器内核的性能。您可以通过将参数值设为 **Disabled** 来禁用。如果您在某些集群机器上禁用并发多线程，则必须在所有集群机器上禁用。

重要

如果禁用并发多线程，请确保在容量规划时考虑到机器性能可能会显著降低的问题。如果您禁用并发多线程，请为您的机器使用较大的类型，如 **n1-standard-8**。

7 指定虚拟机实例所在的主项目。

8 指定 VPC 网络所在区域。

9 是否启用或禁用 FIPS 模式。默认情况下不启用 FIPS 模式。如果启用了 FIPS 模式，运行 OpenShift Container Platform 的 Red Hat Enterprise Linux CoreOS (RHCOS) 机器会绕过默认的 Kubernetes 加密套件，并使用由 RHCOS 提供的加密模块。

重要

只有在 **x86_64** 架构中的 OpenShift Container Platform 部署支持 FIPS 验证的/Modules in Process 加密库。

10 您可以选择提供您用来访问集群中机器的 **sshKey** 值。

注意

对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 **ssh-agent** 进程使用的 SSH 密钥。

11 如何发布集群的面向用户的端点。把 **publish** 设置为 **Internal** 以部署一个私有集群，它不能被互联网访问。默认值为 **External**。要在使用您置备的基础架构的集群中使用共享 VPC，必须将 **publish** 设置为 **Internal**。安装程序将不再能够访问主机项目中的基域的公共 DNS 区域。

4.10.6.3. 在安装过程中配置集群范围代理

生产环境可能会拒绝直接访问互联网，而是提供 HTTP 或 HTTPS 代理。您可以通过在 **install-config.yaml** 文件中配置代理设置，将新的 OpenShift Container Platform 集群配置为使用代理。

先决条件

- 您有一个现有的 **install-config.yaml** 文件。
- 您检查了集群需要访问的站点，并决定是否需要绕过代理。默认情况下代理所有集群出口流量，包括对托管云供应商 API 的调用。您需要将站点添加到 **Proxy** 对象的 **spec.noProxy** 字段来绕过代理。

注意

Proxy 对象 **status.noProxy** 字段使用安装配置中的 **networking.machineNetwork[].cidr**、**networking.clusterNetwork[].cidr** 和 **networking.serviceNetwork[]** 字段的值填充。

对于在 Amazon Web Services(AWS)、Google Cloud Platform(GCP)、Microsoft Azure 和 Red Hat OpenStack Platform(RHOSP)上安装，**Proxy** 对象 **status.noProxy** 字段也会使用实例元数据端点填充(**169.254.169.254**)。

流程

1. 编辑 **install-config.yaml** 文件并添加代理设置。例如：

```
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: example.com 3
additionalTrustBundle: | 4
    -----BEGIN CERTIFICATE-----
```

```
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
...
```

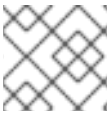
1. 用于创建集群外 HTTP 连接的代理 URL。URL 必须是 **http**。
2. 用于创建集群外 HTTPS 连接的代理 URL。
3. 要排除在代理中的目标域名、IP 地址或其他网络 CIDR 的逗号分隔列表。在域前面加 **.** 来仅匹配子域。例如： **.y.com** 匹配 **x.y.com**，但不匹配 **y.com**。使用 ***** 绕过所有目的地的代理。
4. 如果提供，安装程序会在 **openshift-config 命名空间中生成名为 user-ca- bundle** 的配置映射来保存额外的 CA 证书。如果您提供 **additionalTrustBundle** 和至少一个代理设置，则 **Proxy** 对象会被配置为引用 **trustedCA** 字段中的 **user-ca-bundle** 配置映射。然后，Cluster Network Operator 会创建一个 **trusted-ca-bundle** 配置映射，该配置映射将为 **trustedCA** 参数指定的内容与 RHCOS 信任捆绑包合并。 **additionalTrustBundle** 字段是必需的，除非代理的身份证书由来自 RHCOS 信任捆绑包的颁发机构签名。

注意

安装程序不支持代理的 **readinessEndpoints** 字段。

2. 保存该文件，并在安装 OpenShift Container Platform 时引用。

安装程序会创建一个名为 **cluster** 的集群范围代理，该代理使用提供的 **install-config.yaml** 文件中的代理设置。如果没有提供代理设置，仍然会创建一个 **cluster Proxy** 对象，但它会有一个空 **spec**。



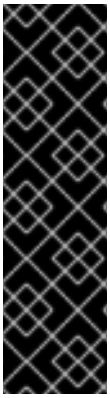
注意

只支持名为 **cluster** 的 **Proxy** 对象，且无法创建额外的代理。

4.10.6.4. 创建 Kubernetes 清单和 Ignition 配置文件

由于您必须修改一些集群定义文件并要手动启动集群机器，因此您必须生成 Kubernetes 清单和 Ignition 配置文件，集群需要这两项来创建其机器。

安装配置文件转换为 Kubernetes 清单。清单嵌套到 Ignition 配置文件中，稍后用于创建集群。



重要

- 安装程序生成的 Ignition 配置文件包含在 24 小时后过期的证书，然后在过期时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外情况是，您需要手动批准待处理的 **node-bootstrapper** 证书签名请求（CSR）来恢复 kubelet 证书。如需更多信息，请参阅 *从过期的 control plane 证书中恢复* 的文档。
- 建议您在生成 12 小时后使用 Ignition 配置文件，因为集群安装后 24 小时证书从 16 小时轮转至 22 小时。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中运行证书更新时避免安装失败。

先决条件

- 已获得 OpenShift Container Platform 安装程序。

- 已创建 **install-config.yaml** 安装配置文件。

流程

1. 切换到包含安装程序的目录，并为集群生成 Kubernetes 清单：

```
$ ./openshift-install create manifests --dir <installation_directory> 1
```

1 对于 **<installation_directory>**，请指定含有您创建的 **install-config.yaml** 文件的安装目录。

2. 删除定义 control plane 机器的 Kubernetes 清单文件：

```
$ rm -f <installation_directory>/openshift/99_openshift-cluster-api_master-machines-*.yaml
```

通过删除这些文件，您可以防止集群自动生成 control plane 机器。

3. 删除定义 worker 机器的 Kubernetes 清单文件：

```
$ rm -f <installation_directory>/openshift/99_openshift-cluster-api_worker-machineset-*.yaml
```

由于您要自行创建并管理 worker 机器，因此不需要初始化这些机器。

4. 检查 **<installation_directory>/manifests/cluster-scheduler-02-config.yml** Kubernetes 清单文件中的 **mastersSchedulable** 参数是否已设置为 **false**。此设置可防止在 control plane 机器上调度 pod:

   a. 打开 **<installation_directory>/manifests/cluster-scheduler-02-config.yml** 文件。

   b. 找到 **mastersSchedulable** 参数并确保它被设置为 **false**。

   c. 保存并退出文件。

5. 删除 **<installation_directory>/manifests/cluster-dns-02-config.yml** DNS 配置文件中的 privateZone **部分**:

```
apiVersion: config.openshift.io/v1
kind: DNS
metadata:
  creationTimestamp: null
  name: cluster
spec:
  baseDomain: example.openshift.com
  privateZone: 1
    id: mycluster-100419-private-zone
status: {}
```

1 完全删除此部分。

6. 配置 VPC 的云供应商。

   a. 打开 **<installation_directory>/manifests/cloud-provider-config.yaml** 文件。

   b. 添加 **network-project-id** 参数，并将其值设为托管共享 VPC 网络的项目 ID。

c. 添加 **network-name** 参数，将其值设置为托管 OpenShift Container Platform 集群的共享 VPC 网络的名称。

d. 将 **subnetwork-name** 参数的值替换为托管计算机器的共享 VPC 子网的值。

<installation_directory>/manifests/cloud-provider-config.yaml 的内容类似以下示例：

```
config: |+
  [global]
  project-id      = example-project
  regional        = true
  multizone       = true
  node-tags       = opensh-ptzzx-master
  node-tags       = opensh-ptzzx-worker
  node-instance-prefix = opensh-ptzzx
  external-instance-groups-prefix = opensh-ptzzx
  network-project-id = example-shared-vpc
  network-name    = example-network
  subnetwork-name = example-worker-subnet
```

7. 如果您部署了不属于私有网络的集群，打开 **<installation_directory>/manifests/cluster-ingress-default-ingresscontroller.yaml** 文件，将 **scope** 参数的值替换为 **External**。该文件类似于以下示例：

```
apiVersion: operator.openshift.io/v1
kind: IngressController
metadata:
  creationTimestamp: null
  name: default
  namespace: openshift-ingress-operator
spec:
  endpointPublishingStrategy:
    loadBalancer:
      scope: External
    type: LoadBalancerService
status:
  availableReplicas: 0
  domain: ''
  selector: ''
```

8. 要创建 Ignition 配置文件，从包含安装程序的目录运行以下命令：

```
$ ./openshift-install create ignition-configs --dir <installation_directory> 1
```

1 对于 **<installation_directory>**，请指定相同的安装目录。

该目录中将生成以下文件：

```
.
├── auth
│   ├── kubeadmin-password
│   └── kubeconfig
├── bootstrap.ign
```

```
├── master.ign
├── metadata.json
└── worker.ign
```

4.10.7. 导出常用变量

4.10.7.1. 提取基础架构名称

Ignition 配置文件包含一个唯一集群标识符，您可以使用它在 Google Cloud Platform (GCP) 中唯一地标识您的集群。基础架构名称还用于在 OpenShift Container Platform 安装过程中定位适当的 GCP 资源。提供的 Deployment Manager 模板包含对此基础架构名称的引用，因此您必须提取它。

先决条件

- 获取 OpenShift Container Platform 安装程序以及集群的 pull secret。
- 已为集群生成 Ignition 配置文件。
- 安装了 **jq** 软件包。

流程

- 要从 Ignition 配置文件元数据中提取和查看基础架构名称，请运行以下命令：

```
$ jq -r .infraID <installation_directory>/metadata.json 1
```

1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

输出示例

```
openshift-vw9j6 1
```

1 此命令的输出是您的集群名称和随机字符串。

4.10.7.2. 为 Deployment Manager 模板导出常用变量

您必须导出与提供的 Deployment Manager 模板搭配使用的一组常用变量，它们有助于在 Google Cloud Platform (GCP) 上完成用户提供基础架构安装。

注意

特定的 Deployment Manager 模板可能还需要其他导出变量，这些变量在相关的程序中详细介绍。

先决条件

- 获取 OpenShift Container Platform 安装程序以及集群的 pull secret。
- 为集群生成 Ignition 配置文件。
- 安装 **jq** 软件包。

流程

1. 导出由提供的 Deployment Manager 模板使用的以下常用变量：

```
$ export BASE_DOMAIN='<base_domain>' 1
$ export BASE_DOMAIN_ZONE_NAME='<base_domain_zone_name>' 2
$ export NETWORK_CIDR='10.0.0.0/16'

$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 3
$ export CLUSTER_NAME=`jq -r .clusterName <installation_directory>/metadata.json`
$ export INFRA_ID=`jq -r .infraID <installation_directory>/metadata.json`
$ export PROJECT_NAME=`jq -r .gcp.projectID <installation_directory>/metadata.json`
```

1 2 提供主机项目的值。

3 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

4.10.8. 用户置备的基础架构对网络的要求

所有 Red Hat Enterprise Linux CoreOS（RHCOS）机器在启动过程中需要 **initramfs** 中的网络从机器配置服务器获取 Ignition 配置。

您必须配置机器间的网络连接，以便集群组件进行通信。每台机器都必须能够解析集群中所有其他机器的主机名。

表 4.40. 所有机器到所有机器

协议	端口	描述
ICMP	N/A	网络可访问性测试
TCP	**1936**	指标
	9000-9999	主机级别的服务，包括端口 **9100-9101** 上的节点导出器和端口 **9099** 上的 Cluster Version Operator。
	10250-10259	Kubernetes 保留的默认端口
	10256	openshift-sdn
UDP	**4789**	VXLAN 和 Geneve
	6081	VXLAN 和 Geneve
	9000-9999	主机级别的服务，包括端口 **9100-9101** 上的节点导出器。
TCP/UDP	**30000-32767**	Kubernetes 节点端口

表 4.41. 要通过控制平面的所有机器

协议	端口	描述
TCP	**6443**	Kubernetes API

表 4.42. control plane 机器到 control plane 机器

协议	端口	描述
TCP	**2379-2380**	etcd 服务器和对等端口

网络拓扑要求

您为集群置备的基础架构必须满足下列网络拓扑要求。

重要

OpenShift Container Platform 要求所有节点都能访问互联网，以便为平台容器提取镜像并向红帽提供遥测数据。

负载均衡器

在安装 OpenShift Container Platform 前，您必须置备两个满足以下要求的负载均衡器：

1. **API 负载均衡器**：提供一个通用端点，供用户（包括人和机器）与平台交互和配置。配置以下条件：
   - 只适用于第 4 层负载均衡。这可被称为 Raw TCP、SSL Passthrough 或者 SSL 桥接模式。如果使用 SSL Bridge 模式，必须为 API 路由启用 Server Name Indication（SNI）。
   - 无状态负载平衡算法。这些选项根据负载均衡器的实现而有所不同。

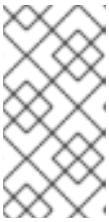
重要

不要为 API 负载均衡器配置会话持久性。

在负载均衡器的前端和后台配置以下端口：

表 4.43. API 负载均衡器

端口	后端机器（池成员）	内部	外部	描述
6443	Bootstrap 和 control plane.bootstrap 机器初始化集群 control plane 后，您要从负载均衡器中删除 bootstrap 机器。您必须为 API 服务器健康检查探测配置 **/readyz** 端点。	X	X	Kubernetes API 服务器
22623	Bootstrap 和 control plane.bootstrap 机器初始化集群 control plane 后，您要从负载均衡器中删除 bootstrap 机器。	X		机器配置服务器



注意

负载均衡器必须配置为，从 API 服务器关闭 **/readyz** 端点到从池中删除 API 服务器实例时最多需要 30 秒。在 **/readyz** 返回错误或处于健康状态后的时间范围内，端点必须被删除或添加。每 5 秒或 10 秒探测一次，有两个成功请求处于健康状态，三个成为不健康的请求经过测试。

2. **应用程序入口负载均衡器**:提供来自集群外部的应用程序流量流量的 Ingress 点。配置以下条件：

   - 只适用于第 4 层负载均衡。这可被称为 Raw TCP、SSL Passthrough 或者 SSL 桥接模式。如果使用 SSL Bridge 模式，您必须为 Ingress 路由启用Server Name Indication（SNI）。
   - 建议根据可用选项以及平台上托管的应用程序类型，使用基于连接的或者基于会话的持久性。

   在负载均衡器的前端和后台配置以下端口：

表 4.44. 应用程序入口负载均衡器

端口	后端机器（池成员）	内部	外部	描述
443	默认运行入口路由器 Pod、计算或 worker 的机器。	X	X	HTTPS 流量
80	默认运行入口路由器 Pod、计算或 worker 的机器。	X	X	HTTP 流量

提示

如果负载均衡器可以看到客户端的真实 IP 地址，启用基于 IP 的会话持久性可提高使用端到端 TLS 加密的应用程序的性能。

注意

OpenShift Container Platform 集群需要正确配置入口路由器。control plane 初始化后，您必须配置入口路由器。

4.10.9. 在 GCP 中创建负载均衡器

您必须在 Google Cloud Platform (GCP) 中配置负载均衡器，供您的 OpenShift Container Platform 集群使用。创建这些组件的一种方法是修改提供的 Deployment Manager 模板。

注意

如果不使用提供的 Deployment Manager 模板来创建 GCP 基础架构，您必须检查提供的信息并手动创建基础架构。如果集群没有正确初始化，您可能需要联系红帽支持并提供您的安装日志。

先决条件

- 配置 GCP 帐户。
- 为集群生成 Ignition 配置文件。

- 在 GCP 中创建和配置 VPC 及相关子网。

流程

1. 复制负载**均衡器的 Deployment Manager 模板**一节中的模板，并将它以 **02_lb_int.py** 形式保存到计算机上。此模板描述了集群所需的负载均衡对象。

2. 对于外部集群，还要将本主题的 **外部负载均衡器的 Deployment Manager 模板** 部分中的模板进行复制，并将它以 **02_lb_ext.py** 形式保存到计算机上。此模板描述了集群所需的外部负载均衡对象。

3. 导出部署模板使用的变量：

   a. 导出集群网络位置：

   ```
   $ export CLUSTER_NETWORK=(`gcloud compute networks describe
   ${HOST_PROJECT_NETWORK} --project ${HOST_PROJECT} --account
   ${HOST_PROJECT_ACCOUNT} --format json | jq -r .selfLink`)
   ```

   b. 导出 control plane 子网位置：

   ```
   $ export CONTROL_SUBNET=(`gcloud compute networks subnets describe
   ${HOST_PROJECT_CONTROL_SUBNET} --region=${REGION} --project
   ${HOST_PROJECT} --account ${HOST_PROJECT_ACCOUNT} --format json | jq -r
   .selfLink`)
   ```

   c. 导出集群使用的三个区域（zone）：

   ```
   $ export ZONE_0=(`gcloud compute regions describe ${REGION} --format=json | jq -r
   .zones[0] | cut -d "/" -f9`)
   ```

   ```
   $ export ZONE_1=(`gcloud compute regions describe ${REGION} --format=json | jq -r
   .zones[1] | cut -d "/" -f9`)
   ```

   ```
   $ export ZONE_2=(`gcloud compute regions describe ${REGION} --format=json | jq -r
   .zones[2] | cut -d "/" -f9`)
   ```

4. 创建 **02_infra.yaml** 资源定义文件：

```
$ cat <<EOF >02_infra.yaml
imports:
- path: 02_lb_ext.py
- path: 02_lb_int.py (1)
resources:
- name: cluster-lb-ext (2)
  type: 02_lb_ext.py
  properties:
    infra_id: '${INFRA_ID}' (3)
    region: '${REGION}' (4)
- name: cluster-lb-int
  type: 02_lb_int.py
  properties:
    cluster_network: '${CLUSTER_NETWORK}'
```

```
    control_subnet: '${CONTROL_SUBNET}' (5)
    infra_id: '${INFRA_ID}'
    region: '${REGION}'
    zones: (6)
    - '${ZONE_0}'
    - '${ZONE_1}'
    - '${ZONE_2}'
  EOF
```

(1) (2) 仅在部署外部集群时需要。

(3) **infra_id** 是来自于提取步骤的 **INFRA_ID** 基础架构名称。

(4) **region** 是集群要部署到的区域，如 **us-central1**。

(5) **control_subnet** 是到控制子网的 URI。

(6) **zones** 是 control plane 实例要部署到的区域，如 **us-east1-b**、**us-east1-c** 和 **us-east1-d**。

5. 使用 **gcloud** CLI 创建部署：

```
$ gcloud deployment-manager deployments create ${INFRA_ID}-infra --config 02_infra.yaml
```

6. 导出集群 IP 地址：

```
$ export CLUSTER_IP=(`gcloud compute addresses describe ${INFRA_ID}-cluster-ip --
region=${REGION} --format json | jq -r .address`)
```

7. 对于外部集群，还要导出集群公共 IP 地址：

```
$ export CLUSTER_PUBLIC_IP=(`gcloud compute addresses describe ${INFRA_ID}-cluster-
public-ip --region=${REGION} --format json | jq -r .address`)
```

4.10.9.1. 外部负载均衡器的 Deployment Manager 模板

您可以使用以下 Deployment Manager 模板来部署 OpenShift Container Platform 集群所需的外部负载均衡器：

例 4.11. 02_lb_ext.py Deployment Manager 模板

```
def GenerateConfig(context):

    resources = [{
        'name': context.properties['infra_id'] + '-cluster-public-ip',
        'type': 'compute.v1.address',
        'properties': {
            'region': context.properties['region']
        }
    }, {
        # Refer to docs/dev/kube-apiserver-health-check.md on how to correctly setup health check
probe for kube-apiserver
        'name': context.properties['infra_id'] + '-api-http-health-check',
        'type': 'compute.v1.httpHealthCheck',
```

```
        'properties': {
            'port': 6080,
            'requestPath': '/readyz'
        }
    }, {
        'name': context.properties['infra_id'] + '-api-target-pool',
        'type': 'compute.v1.targetPool',
        'properties': {
            'region': context.properties['region'],
            'healthChecks': ['$(ref.' + context.properties['infra_id'] + '-api-http-health-check.selfLink)'],
            'instances': []
        }
    }, {
        'name': context.properties['infra_id'] + '-api-forwarding-rule',
        'type': 'compute.v1.forwardingRule',
        'properties': {
            'region': context.properties['region'],
            'IPAddress': '$(ref.' + context.properties['infra_id'] + '-cluster-public-ip.selfLink)',
            'target': '$(ref.' + context.properties['infra_id'] + '-api-target-pool.selfLink)',
            'portRange': '6443'
        }
    }]

    return {'resources': resources}
```

4.10.9.2. 内部负载均衡器的 Deployment Manager 模板

您可以使用以下 Deployment Manager 模板来部署 OpenShift Container Platform 集群所需的内部负载均衡器：

例 4.12. 02_lb_int.py Deployment Manager 模板

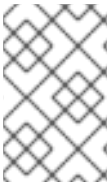
```
def GenerateConfig(context):

    backends = []
    for zone in context.properties['zones']:
        backends.append({
            'group': '$(ref.' + context.properties['infra_id'] + '-master-' + zone + '-instance-group' +
'.selfLink)'
        })

    resources = [{
        'name': context.properties['infra_id'] + '-cluster-ip',
        'type': 'compute.v1.address',
        'properties': {
            'addressType': 'INTERNAL',
            'region': context.properties['region'],
            'subnetwork': context.properties['control_subnet']
        }
    }, {
        # Refer to docs/dev/kube-apiserver-health-check.md on how to correctly setup health check
probe for kube-apiserver
        'name': context.properties['infra_id'] + '-api-internal-health-check',
        'type': 'compute.v1.healthCheck',
```

```
        'properties': {
            'httpsHealthCheck': {
                'port': 6443,
                'requestPath': '/readyz'
            },
            'type': "HTTPS"
        }
    }, {
        'name': context.properties['infra_id'] + '-api-internal-backend-service',
        'type': 'compute.v1.regionBackendService',
        'properties': {
            'backends': backends,
            'healthChecks': ['$(ref.' + context.properties['infra_id'] + '-api-internal-health-
check.selfLink)'],
            'loadBalancingScheme': 'INTERNAL',
            'region': context.properties['region'],
            'protocol': 'TCP',
            'timeoutSec': 120
        }
    }, {
        'name': context.properties['infra_id'] + '-api-internal-forwarding-rule',
        'type': 'compute.v1.forwardingRule',
        'properties': {
            'backendService': '$(ref.' + context.properties['infra_id'] + '-api-internal-backend-
service.selfLink)',
            'IPAddress': '$(ref.' + context.properties['infra_id'] + '-cluster-ip.selfLink)',
            'loadBalancingScheme': 'INTERNAL',
            'ports': ['6443','22623'],
            'region': context.properties['region'],
            'subnetwork': context.properties['control_subnet']
        }
    }]

    for zone in context.properties['zones']:
        resources.append({
            'name': context.properties['infra_id'] + '-master-' + zone + '-instance-group',
            'type': 'compute.v1.instanceGroup',
            'properties': {
                'namedPorts': [
                    {
                        'name': 'ignition',
                        'port': 22623
                    }, {
                        'name': 'https',
                        'port': 6443
                    }
                ],
                'network': context.properties['cluster_network'],
                'zone': zone
            }
        })

    return {'resources': resources}
```

创建外部集群时，除了 **02_lb_ext.py** 模板外，还需要此模板。

4.10.10. 在 GCP 中创建私有 DNS 区域

您必须在 Google Cloud Platform (GCP) 中配置一个私人的 DNS 区以供您的 OpenShift Container Platform 集群使用。创建此组件的一种方法是修改提供的 Deployment Manager 模板。



注意

如果不使用提供的 Deployment Manager 模板来创建 GCP 基础架构，您必须检查提供的信息并手动创建基础架构。如果集群没有正确初始化，您可能需要联系红帽支持并提供您的安装日志。

先决条件

- 配置 GCP 帐户。
- 为集群生成 Ignition 配置文件。
- 在 GCP 中创建和配置 VPC 及相关子网。

流程

1. 复制 **私人 DNS 的 Deployment Manager 模板**一节中的模板，并将它以 **02_dns.py** 形式保存到计算机上。此模板描述了集群所需的私有 DNS 对象。
2. 创建 **02_dns.yaml** 资源定义文件：

```
$ cat <<EOF >02_dns.yaml
imports:
- path: 02_dns.py

resources:
- name: cluster-dns
  type: 02_dns.py
  properties:
    infra_id: '${INFRA_ID}' (1)
    cluster_domain: '${CLUSTER_NAME}.${BASE_DOMAIN}' (2)
    cluster_network: '${CLUSTER_NETWORK}' (3)
EOF
```

(1) **infra_id** 是来自于提取步骤的 **INFRA_ID** 基础架构名称。

(2) **cluster_domain** 是集群的域，如 **openshift.example.com**。

(3) **cluster_network** 是集群网络的 **selfLink** URL。

3. 使用 **gcloud** CLI 创建部署：

```
$ gcloud deployment-manager deployments create ${INFRA_ID}-dns --config 02_dns.yaml --
project ${HOST_PROJECT} --account ${HOST_PROJECT_ACCOUNT}
```

4. 由于 Deployment Manager 的限制，模板不会创建 DNS 条目，因此您必须手动创建它们：
   a. 添加内部 DNS 条目：

```
$ if [ -f transaction.yaml ]; then rm transaction.yaml; fi
$ gcloud dns record-sets transaction start --zone ${INFRA_ID}-private-zone --project
${HOST_PROJECT} --account ${HOST_PROJECT_ACCOUNT}
$ gcloud dns record-sets transaction add ${CLUSTER_IP} --name
api.${CLUSTER_NAME}.${BASE_DOMAIN}. --ttl 60 --type A --zone ${INFRA_ID}-
private-zone --project ${HOST_PROJECT} --account ${HOST_PROJECT_ACCOUNT}
$ gcloud dns record-sets transaction add ${CLUSTER_IP} --name api-
int.${CLUSTER_NAME}.${BASE_DOMAIN}. --ttl 60 --type A --zone ${INFRA_ID}-
private-zone --project ${HOST_PROJECT} --account ${HOST_PROJECT_ACCOUNT}
$ gcloud dns record-sets transaction execute --zone ${INFRA_ID}-private-zone --project
${HOST_PROJECT} --account ${HOST_PROJECT_ACCOUNT}
```

b. 对于外部集群，还要添加外部 DNS 条目：

```
$ if [ -f transaction.yaml ]; then rm transaction.yaml; fi
$ gcloud --account=${HOST_PROJECT_ACCOUNT} --project=${HOST_PROJECT} dns
record-sets transaction start --zone ${BASE_DOMAIN_ZONE_NAME}
$ gcloud --account=${HOST_PROJECT_ACCOUNT} --project=${HOST_PROJECT} dns
record-sets transaction add ${CLUSTER_PUBLIC_IP} --name
api.${CLUSTER_NAME}.${BASE_DOMAIN}. --ttl 60 --type A --zone
${BASE_DOMAIN_ZONE_NAME}
$ gcloud --account=${HOST_PROJECT_ACCOUNT} --project=${HOST_PROJECT} dns
record-sets transaction execute --zone ${BASE_DOMAIN_ZONE_NAME}
```

4.10.10.1. 私有 DNS 的 Deployment Manager 模板

您可以使用以下 Deployment Manager 模板来部署 OpenShift Container Platform 集群所需的 私有 DNS：

例 4.13. 02_dns.py Deployment Manager 模板

```
def GenerateConfig(context):

    resources = [{
        'name': context.properties['infra_id'] + '-private-zone',
        'type': 'dns.v1.managedZone',
        'properties': {
            'description': '',
            'dnsName': context.properties['cluster_domain'] + '.',
            'visibility': 'private',
            'privateVisibilityConfig': {
                'networks': [{
                    'networkUrl': context.properties['cluster_network']
                }]
            }
        }
    }]

    return {'resources': resources}
```

4.10.11. 在 GCP 中创建防火墙规则

您必须在 Google Cloud Platform (GCP) 中创建一个防火墙，供您的 OpenShift Container Platform 集群使用。创建这些组件的一种方法是修改提供的 Deployment Manager 模板。

注意

如果不使用提供的 Deployment Manager 模板来创建 GCP 基础架构，您必须检查提供的信息并手动创建基础架构。如果集群没有正确初始化，您可能需要联系红帽支持并提供您的安装日志。

先决条件

- 配置 GCP 帐户。
- 为集群生成 Ignition 配置文件。
- 在 GCP 中创建和配置 VPC 及相关子网。

流程

1. 复制**防火墙规则的 Deployment Manager 模板**一节中的模板，并将它以 **03_firewall.py** 形式保存到计算机上。此模板描述了集群所需的安全组。
2. 创建 **03_firewall.yaml** 资源定义文件：

```
$ cat <<EOF >03_firewall.yaml
imports:
- path: 03_firewall.py

resources:
- name: cluster-firewall
  type: 03_firewall.py
  properties:
    allowed_external_cidr: '0.0.0.0/0' (1)
    infra_id: '${INFRA_ID}' (2)
    cluster_network: '${CLUSTER_NETWORK}' (3)
    network_cidr: '${NETWORK_CIDR}' (4)
EOF
```

(1) **allowed_external_cidr** 是 CIDR 范围，可访问集群 API 和 SSH 到 bootstrap 主机。对于内部集群，将此值设置为 **${NETWORK_CIDR}**。

(2) **infra_id** 是来自于提取步骤的 **INFRA_ID** 基础架构名称。

(3) **cluster_network** 是集群网络的 **selfLink** URL。

(4) **network_cidr** 是 VPC 网络的 CIDR，如 **10.0.0.0/16**。

3. 使用 **gcloud** CLI 创建部署：

```
$ gcloud deployment-manager deployments create ${INFRA_ID}-firewall --config
03_firewall.yaml --project ${HOST_PROJECT} --account ${HOST_PROJECT_ACCOUNT}
```

4.10.11.1. 防火墙规则的 Deployment Manager 模板

您可以使用以下 Deployment Manager 模板来部署 OpenShift Container Platform 集群所需的防火墙规则：

例 4.14. 03_firewall.py Deployment Manager 模板

```
def GenerateConfig(context):

    resources = [{
        'name': context.properties['infra_id'] + '-bootstrap-in-ssh',
        'type': 'compute.v1.firewall',
        'properties': {
            'network': context.properties['cluster_network'],
            'allowed': [{
                'IPProtocol': 'tcp',
                'ports': ['22']
            }],
            'sourceRanges': [context.properties['allowed_external_cidr']],
            'targetTags': [context.properties['infra_id'] + '-bootstrap']
        }
    }, {
        'name': context.properties['infra_id'] + '-api',
        'type': 'compute.v1.firewall',
        'properties': {
            'network': context.properties['cluster_network'],
            'allowed': [{
                'IPProtocol': 'tcp',
                'ports': ['6443']
            }],
            'sourceRanges': [context.properties['allowed_external_cidr']],
            'targetTags': [context.properties['infra_id'] + '-master']
        }
    }, {
        'name': context.properties['infra_id'] + '-health-checks',
        'type': 'compute.v1.firewall',
        'properties': {
            'network': context.properties['cluster_network'],
            'allowed': [{
                'IPProtocol': 'tcp',
                'ports': ['6080', '6443', '22624']
            }],
            'sourceRanges': ['35.191.0.0/16', '130.211.0.0/22', '209.85.152.0/22', '209.85.204.0/22'],
            'targetTags': [context.properties['infra_id'] + '-master']
        }
    }, {
        'name': context.properties['infra_id'] + '-etcd',
        'type': 'compute.v1.firewall',
        'properties': {
            'network': context.properties['cluster_network'],
            'allowed': [{
                'IPProtocol': 'tcp',
                'ports': ['2379-2380']
            }],
            'sourceTags': [context.properties['infra_id'] + '-master'],
            'targetTags': [context.properties['infra_id'] + '-master']
        }
    }, {
```

```
        'name': context.properties['infra_id'] + '-control-plane',
        'type': 'compute.v1.firewall',
        'properties': {
            'network': context.properties['cluster_network'],
            'allowed': [{
                'IPProtocol': 'tcp',
                'ports': ['10257']
            },{
                'IPProtocol': 'tcp',
                'ports': ['10259']
            },{
                'IPProtocol': 'tcp',
                'ports': ['22623']
            }],
            'sourceTags': [
                context.properties['infra_id'] + '-master',
                context.properties['infra_id'] + '-worker'
            ],
            'targetTags': [context.properties['infra_id'] + '-master']
        }
    }, {
        'name': context.properties['infra_id'] + '-internal-network',
        'type': 'compute.v1.firewall',
        'properties': {
            'network': context.properties['cluster_network'],
            'allowed': [{
                'IPProtocol': 'icmp'
            },{
                'IPProtocol': 'tcp',
                'ports': ['22']
            }],
            'sourceRanges': [context.properties['network_cidr']],
            'targetTags': [
                context.properties['infra_id'] + '-master',
                context.properties['infra_id'] + '-worker'
            ]
        }
    }, {
        'name': context.properties['infra_id'] + '-internal-cluster',
        'type': 'compute.v1.firewall',
        'properties': {
            'network': context.properties['cluster_network'],
            'allowed': [{
                'IPProtocol': 'udp',
                'ports': ['4789', '6081']
            },{
                'IPProtocol': 'tcp',
                'ports': ['9000-9999']
            },{
                'IPProtocol': 'udp',
                'ports': ['9000-9999']
            },{
                'IPProtocol': 'tcp',
                'ports': ['10250']
            },{
                'IPProtocol': 'tcp',
```

```
            'ports': ['30000-32767']
        },{
            'IPProtocol': 'udp',
            'ports': ['30000-32767']
        }],
        'sourceTags': [
            context.properties['infra_id'] + '-master',
            context.properties['infra_id'] + '-worker'
        ],
        'targetTags': [
            context.properties['infra_id'] + '-master',
            context.properties['infra_id'] + '-worker'
        ]
    }
}]

return {'resources': resources}
```

4.10.12. 在 GCP 中创建 IAM 角色

您必须在 Google Cloud Platform (GCP) 中创建一个 IAM 角色，供您的 OpenShift Container Platform 集群使用。创建这些组件的一种方法是修改提供的 Deployment Manager 模板。

注意

如果不使用提供的 Deployment Manager 模板来创建 GCP 基础架构，您必须检查提供的信息并手动创建基础架构。如果集群没有正确初始化，您可能需要联系红帽支持并提供您的安装日志。

先决条件

- 配置 GCP 帐户。
- 为集群生成 Ignition 配置文件。
- 在 GCP 中创建和配置 VPC 及相关子网。

流程

1. 复制 **IAM 角色的 Deployment Manager 模板**一节中的模板，并将它以 **03_iam.p** 形式保存到计算机上。此模板描述了集群所需的 IAM 角色。
2. 创建 **03_iam.yaml** 资源定义文件：

```
$ cat <<EOF >03_iam.yaml
imports:
- path: 03_iam.py
resources:
- name: cluster-iam
  type: 03_iam.py
  properties:
    infra_id: '${INFRA_ID}' (1)
EOF
```

1 **infra_id** 是来自于提取步骤的 **INFRA_ID** 基础架构名称。

3. 使用 **gcloud** CLI 创建部署：

```
$ gcloud deployment-manager deployments create ${INFRA_ID}-iam --config 03_iam.yaml
```

4. 导出 master 服务帐户的变量：

```
$ export MASTER_SERVICE_ACCOUNT=(`gcloud iam service-accounts list --filter
"email~^${INFRA_ID}-m@${PROJECT_NAME}." --format json | jq -r '.[0].email'`)
```

5. 导出 worker 服务帐户的变量：

```
$ export WORKER_SERVICE_ACCOUNT=(`gcloud iam service-accounts list --filter
"email~^${INFRA_ID}-w@${PROJECT_NAME}." --format json | jq -r '.[0].email'`)
```

6. 将安装程序所需的权限分配给托管 control plane 和计算子网的子网的服务帐户：

   a. 为 master 服务账户授予托管共享 VPC 的项目的 **networkViewer** 角色：

```
$ gcloud --account=${HOST_PROJECT_ACCOUNT} --project=${HOST_PROJECT}
projects add-iam-policy-binding ${HOST_PROJECT} --member
"serviceAccount:${MASTER_SERVICE_ACCOUNT}" --role
"roles/compute.networkViewer"
```

   b. 将 **networkUser** 角色授予 control plane 子网的 master 服务帐户：

```
$ gcloud --account=${HOST_PROJECT_ACCOUNT} --project=${HOST_PROJECT}
compute networks subnets add-iam-policy-binding
"${HOST_PROJECT_CONTROL_SUBNET}" --member
"serviceAccount:${MASTER_SERVICE_ACCOUNT}" --role "roles/compute.networkUser"
--region ${REGION}
```

   c. 将 **networkUser** 角色授予 control plane 子网的 worker 服务帐户：

```
$ gcloud --account=${HOST_PROJECT_ACCOUNT} --project=${HOST_PROJECT}
compute networks subnets add-iam-policy-binding
"${HOST_PROJECT_CONTROL_SUBNET}" --member
"serviceAccount:${WORKER_SERVICE_ACCOUNT}" --role
"roles/compute.networkUser" --region ${REGION}
```

   d. 将 **networkUser** 角色授予计算子网的 master 服务帐户：

```
$ gcloud --account=${HOST_PROJECT_ACCOUNT} --project=${HOST_PROJECT}
compute networks subnets add-iam-policy-binding
"${HOST_PROJECT_COMPUTE_SUBNET}" --member
"serviceAccount:${MASTER_SERVICE_ACCOUNT}" --role "roles/compute.networkUser"
--region ${REGION}
```

   e. 将 **networkUser** 角色授予计算子网的 worker 服务帐户：

```
$ gcloud --account=${HOST_PROJECT_ACCOUNT} --project=${HOST_PROJECT}
```

```
compute networks subnets add-iam-policy-binding
"${HOST_PROJECT_COMPUTE_SUBNET}" --member
"serviceAccount:${WORKER_SERVICE_ACCOUNT}" --role
"roles/compute.networkUser" --region ${REGION}
```

7. 由于 Deployment Manager 的限制，模板不会创建策略绑定，因此您必须手动创建它们：

```
$ gcloud projects add-iam-policy-binding ${PROJECT_NAME} --member
"serviceAccount:${MASTER_SERVICE_ACCOUNT}" --role "roles/compute.instanceAdmin"
$ gcloud projects add-iam-policy-binding ${PROJECT_NAME} --member
"serviceAccount:${MASTER_SERVICE_ACCOUNT}" --role "roles/compute.networkAdmin"
$ gcloud projects add-iam-policy-binding ${PROJECT_NAME} --member
"serviceAccount:${MASTER_SERVICE_ACCOUNT}" --role "roles/compute.securityAdmin"
$ gcloud projects add-iam-policy-binding ${PROJECT_NAME} --member
"serviceAccount:${MASTER_SERVICE_ACCOUNT}" --role "roles/iam.serviceAccountUser"
$ gcloud projects add-iam-policy-binding ${PROJECT_NAME} --member
"serviceAccount:${MASTER_SERVICE_ACCOUNT}" --role "roles/storage.admin"

$ gcloud projects add-iam-policy-binding ${PROJECT_NAME} --member
"serviceAccount:${WORKER_SERVICE_ACCOUNT}" --role "roles/compute.viewer"
$ gcloud projects add-iam-policy-binding ${PROJECT_NAME} --member
"serviceAccount:${WORKER_SERVICE_ACCOUNT}" --role "roles/storage.admin"
```

8. 创建服务帐户密钥，并将它保存到本地备用：

```
$ gcloud iam service-accounts keys create service-account-key.json --iam-
account=${MASTER_SERVICE_ACCOUNT}
```

4.10.12.1. IAM 角色的 Deployment Manager 模板

您可以使用以下 Deployment Manager 模板来部署 OpenShift Container Platform 集群所需的 IAM 角色：

例 4.15. 03_iam.py Deployment Manager 模板

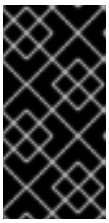
```
def GenerateConfig(context):

    resources = [{
        'name': context.properties['infra_id'] + '-master-node-sa',
        'type': 'iam.v1.serviceAccount',
        'properties': {
            'accountId': context.properties['infra_id'] + '-m',
            'displayName': context.properties['infra_id'] + '-master-node'
        }
    }, {
        'name': context.properties['infra_id'] + '-worker-node-sa',
        'type': 'iam.v1.serviceAccount',
        'properties': {
            'accountId': context.properties['infra_id'] + '-w',
            'displayName': context.properties['infra_id'] + '-worker-node'
        }
    }]

    return {'resources': resources}
```

4.10.13. 为 GCP 基础架构创建 RHCOS 集群镜像

您必须对 OpenShift Container Platform 节点的 Google Cloud Platform (GCP) 使用有效的 Red Hat Enterprise Linux CoreOS (RHCOS) 镜像。

流程

1. 从 RHCOS 镜像镜像页面获取 RHCOS 镜像。



重要

RHCOS 镜像可能不会随着 OpenShift Container Platform 的每一发行版本都有改变。您必须下载一个最高版本的镜像，其版本号应小于或等于您安装的 OpenShift Container Platform 版本。如果可用，请使用与 OpenShift Container Platform 版本匹配的镜像版本。

文件名包含 OpenShift Container Platform 版本号，格式为 **rhcos-<version>-<arch>-gcp.<arch>.tar.gz**。

2. 创建 Google 存储桶：

```
$ gsutil mb gs://<bucket_name>
```

3. 将 RHCOS 镜像上传到 Google 存储桶：

```
$ gsutil cp <downloaded_image_file_path>/rhcos-<version>-x86_64-gcp.x86_64.tar.gz
gs://<bucket_name>
```

4. 将上传的 RHCOS 镜像位置导出为变量：

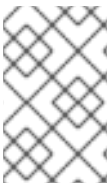
```
$ export IMAGE_SOURCE="gs://<bucket_name>/rhcos-<version>-x86_64-
gcp.x86_64.tar.gz"
```

5. 创建集群镜像：

```
$ gcloud compute images create "${INFRA_ID}-rhcos-image" \
    --source-uri="${IMAGE_SOURCE}"
```

4.10.14. 在 GCP 中创建 bootstrap 机器

您必须在 Google Cloud Platform (GCP) 中创建 bootstrap 机器，以便在 OpenShift Container Platform 集群初始化过程中使用。创建此机器的一种方法是修改提供的 Deployment Manager 模板。



注意

如果不使用提供的 Deployment Manager 模板来创建 bootstrap 机器，您必须检查提供的信息并手动创建基础架构。如果集群没有正确初始化，您可能需要联系红帽支持并提供您的安装日志。

先决条件

- 配置 GCP 帐户。
- 为集群生成 Ignition 配置文件。
- 在 GCP 中创建和配置 VPC 及相关子网。
- 在 GCP 中创建和配置联网及负载均衡器。
- 创建 control plane 和计算角色。
- 确定安装了 pyOpenSSL。

流程

1. 复制 **bootstrap 机器的 Deployment Manager 模板**一节中的模板，并将它以 **04_bootstrap.py** 形式保存到计算机上。此模板描述了集群所需的 bootstrap 机器。
2. 导出安装程序所需的 Red Hat Enterprise Linux CoreOS（RHCOS）镜像的位置：

```
$ export CLUSTER_IMAGE=(`gcloud compute images describe ${INFRA_ID}-rhcos-image --format json | jq -r .selfLink`)
```

3. 创建存储桶并上传 **bootstrap.ign** 文件：

```
$ gsutil mb gs://${INFRA_ID}-bootstrap-ignition
$ gsutil cp <installation_directory>/bootstrap.ign gs://${INFRA_ID}-bootstrap-ignition/
```

4. 为 bootstrap 实例创建一个签名的 URL，用于访问 Ignition 配置。将输出中的 URL 导出为变量：

```
$ export BOOTSTRAP_IGN=`gsutil signurl -d 1h service-account-key.json gs://${INFRA_ID}-bootstrap-ignition/bootstrap.ign | grep "^gs:" | awk '{print $5}'`
```

5. 创建 **04_bootstrap.yaml** 资源定义文件：

```
$ cat <<EOF >04_bootstrap.yaml
imports:
- path: 04_bootstrap.py

resources:
- name: cluster-bootstrap
  type: 04_bootstrap.py
  properties:
    infra_id: '${INFRA_ID}' (1)
    region: '${REGION}' (2)
    zone: '${ZONE_0}' (3)

    cluster_network: '${CLUSTER_NETWORK}' (4)
    control_subnet: '${CONTROL_SUBNET}' (5)
    image: '${CLUSTER_IMAGE}' (6)
    machine_type: 'n1-standard-4' (7)
    root_volume_size: '128' (8)

    bootstrap_ign: '${BOOTSTRAP_IGN}' (9)
EOF
```

1. **infra_id** 是来自于提取步骤的 **INFRA_ID** 基础架构名称。
2. **region** 是集群要部署到的区域，如 **us-central1**。
3. **zone** 是 Bootstrap 实例要部署到的区域，如 **us-central1-b**。
4. **cluster_network** 是集群网络的 **selfLink** URL。
5. **control_subnet** 是控制子网的 **selfLink** URL。
6. **image** 是 RHCOS 镜像的 **selfLink** URL。
7. **machine_type** 是实例的机器类型，如 **n1-standard-4**。
8. **root_volume_size** 是 bootstrap 机器的引导磁盘大小。
9. **bootstrap_ign** 是创建签名 URL 时的 URL 输出。

6. 使用 **gcloud** CLI 创建部署：

```
$ gcloud deployment-manager deployments create ${INFRA_ID}-bootstrap --config
04_bootstrap.yaml
```

7. 将 bootstrap 实例添加到内部负载均衡器实例组中：

```
$ gcloud compute instance-groups unmanaged add-instances ${INFRA_ID}-bootstrap-
instance-group --zone=${ZONE_0} --instances=${INFRA_ID}-bootstrap
```

8. 将 bootstrap 实例组添加到内部负载均衡器后端服务中：

```
$ gcloud compute backend-services add-backend ${INFRA_ID}-api-internal-backend-service
--region=${REGION} --instance-group=${INFRA_ID}-bootstrap-instance-group --instance-
group-zone=${ZONE_0}
```

4.10.14.1. bootstrap 机器的 Deployment Manager 模板

您可以使用以下 Deployment Manager 模板来部署 OpenShift Container Platform 集群所需的 bootstrap 机器：

例 4.16. 04_bootstrap.py Deployment Manager 模板

```
def GenerateConfig(context):

  resources = [{
    'name': context.properties['infra_id'] + '-bootstrap-public-ip',
    'type': 'compute.v1.address',
    'properties': {
        'region': context.properties['region']
    }
  }, {
    'name': context.properties['infra_id'] + '-bootstrap',
    'type': 'compute.v1.instance',
    'properties': {
```

```
            'disks': [{
                'autoDelete': True,
                'boot': True,
                'initializeParams': {
                    'diskSizeGb': context.properties['root_volume_size'],
                    'sourceImage': context.properties['image']
                }
            }],
            'machineType': 'zones/' + context.properties['zone'] + '/machineTypes/' +
context.properties['machine_type'],
            'metadata': {
                'items': [{
                    'key': 'user-data',
                    'value': '{"ignition":{"config":{"replace":{"source":"' + context.properties['bootstrap_ign']
+ '"}},"version":"3.1.0"}}',
                }]
            },
            'networkInterfaces': [{
                'subnetwork': context.properties['control_subnet'],
                'accessConfigs': [{
                    'natIP': '$(ref.' + context.properties['infra_id'] + '-bootstrap-public-ip.address)'
                }]
            }],
            'tags': {
                'items': [
                    context.properties['infra_id'] + '-master',
                    context.properties['infra_id'] + '-bootstrap'
                ]
            },
            'zone': context.properties['zone']
        }
    }, {
        'name': context.properties['infra_id'] + '-bootstrap-instance-group',
        'type': 'compute.v1.instanceGroup',
        'properties': {
            'namedPorts': [
                {
                    'name': 'ignition',
                    'port': 22623
                }, {
                    'name': 'https',
                    'port': 6443
                }
            ],
            'network': context.properties['cluster_network'],
            'zone': context.properties['zone']
        }
    }]

    return {'resources': resources}
```

4.10.15. 在 GCP 中创建 control plane 机器

您必须在 Google Cloud Platform (GCP) 中创建 control plane 机器，供您的集群使用。创建这些机器的一种方法是修改提供的 Deployment Manager 模板。

注意

如果不使用提供的 Deployment Manager 模板来创建 control plane 机器，您必须检查提供的信息并手动创建基础架构。如果集群没有正确初始化，您可能需要联系红帽支持并提供您的安装日志。

先决条件

- 配置 GCP 帐户。
- 为集群生成 Ignition 配置文件。
- 在 GCP 中创建和配置 VPC 及相关子网。
- 在 GCP 中创建和配置联网及负载均衡器。
- 创建 control plane 和计算角色。
- 创建 bootstrap 机器。

流程

1. 复制 **control plane 机器的 Deployment Manager 模板**一节中的模板，并将它以 **05_control_plane.py** 形式保存到计算机上。此模板描述了集群所需的 control plane 机器。
2. 导出资源定义所需的以下变量：

```
$ export MASTER_IGNITION=`cat <installation_directory>/master.ign`
```

3. 创建 **05_control_plane.yaml** 资源定义文件：

```
$ cat <<EOF >05_control_plane.yaml
imports:
- path: 05_control_plane.py

resources:
- name: cluster-control-plane
  type: 05_control_plane.py
  properties:
    infra_id: '${INFRA_ID}' 1
    zones: 2
    - '${ZONE_0}'
    - '${ZONE_1}'
    - '${ZONE_2}'

    control_subnet: '${CONTROL_SUBNET}' 3
    image: '${CLUSTER_IMAGE}' 4
    machine_type: 'n1-standard-4' 5
    root_volume_size: '128'
    service_account_email: '${MASTER_SERVICE_ACCOUNT}' 6
```

```
  ignition: '${MASTER_IGNITION}' 7
EOF
```

1. **infra_id** 是来自于提取步骤的 **INFRA_ID** 基础架构名称。
2. **zones** 是 control plane 实例要部署到的区域，如 **us-central1-a**、**us-central1-b** 和 **us-central1-c**。
3. **control_subnet** 是控制子网的 **selfLink** URL。
4. **image** 是 RHCOS 镜像的 **selfLink** URL。
5. **machine_type** 是实例的机器类型，如 **n1-standard-4**。
6. **service_account_email** 是创建的 master 服务帐户的电子邮件地址。
7. **ignition** 是 **master.ign** 文件的内容。

4. 使用 **gcloud** CLI 创建部署：

```
$ gcloud deployment-manager deployments create ${INFRA_ID}-control-plane --config
05_control_plane.yaml
```

5. 由于 Deployment Manager 的限制，模板无法管理负载均衡器成员资格，因此您必须手动添加 control plane 机器。
   - 运行以下命令，将 control plane 机器添加到适当的实例组中：

```
$ gcloud compute instance-groups unmanaged add-instances ${INFRA_ID}-master-
${ZONE_0}-instance-group --zone=${ZONE_0} --instances=${INFRA_ID}-master-0
$ gcloud compute instance-groups unmanaged add-instances ${INFRA_ID}-master-
${ZONE_1}-instance-group --zone=${ZONE_1} --instances=${INFRA_ID}-master-1
$ gcloud compute instance-groups unmanaged add-instances ${INFRA_ID}-master-
${ZONE_2}-instance-group --zone=${ZONE_2} --instances=${INFRA_ID}-master-2
```

   - 对于外部集群，还需要运行以下命令将 control plane 机器添加到目标池中：

```
$ gcloud compute target-pools add-instances ${INFRA_ID}-api-target-pool --instances-
zone="${ZONE_0}" --instances=${INFRA_ID}-master-0
$ gcloud compute target-pools add-instances ${INFRA_ID}-api-target-pool --instances-
zone="${ZONE_1}" --instances=${INFRA_ID}-master-1
$ gcloud compute target-pools add-instances ${INFRA_ID}-api-target-pool --instances-
zone="${ZONE_2}" --instances=${INFRA_ID}-master-2
```

4.10.15.1. control plane 机器的 Deployment Manager 模板

您可以使用以下 Deployment Manager 模板来部署 OpenShift Container Platform 集群所需的control plane 机器：

例 4.17. 05_control_plane.py Deployment Manager 模板

```
def GenerateConfig(context):
```

```
    resources = [{
        'name': context.properties['infra_id'] + '-master-0',
        'type': 'compute.v1.instance',
        'properties': {
            'disks': [{
                'autoDelete': True,
                'boot': True,
                'initializeParams': {
                    'diskSizeGb': context.properties['root_volume_size'],
                    'diskType': 'zones/' + context.properties['zones'][0] + '/diskTypes/pd-ssd',
                    'sourceImage': context.properties['image']
                }
            }],
            'machineType': 'zones/' + context.properties['zones'][0] + '/machineTypes/' +
context.properties['machine_type'],
            'metadata': {
                'items': [{
                    'key': 'user-data',
                    'value': context.properties['ignition']
                }]
            },
            'networkInterfaces': [{
                'subnetwork': context.properties['control_subnet']
            }],
            'serviceAccounts': [{
                'email': context.properties['service_account_email'],
                'scopes': ['https://www.googleapis.com/auth/cloud-platform']
            }],
            'tags': {
                'items': [
                    context.properties['infra_id'] + '-master',
                ]
            },
            'zone': context.properties['zones'][0]
        }
    }, {
        'name': context.properties['infra_id'] + '-master-1',
        'type': 'compute.v1.instance',
        'properties': {
            'disks': [{
                'autoDelete': True,
                'boot': True,
                'initializeParams': {
                    'diskSizeGb': context.properties['root_volume_size'],
                    'diskType': 'zones/' + context.properties['zones'][1] + '/diskTypes/pd-ssd',
                    'sourceImage': context.properties['image']
                }
            }],
            'machineType': 'zones/' + context.properties['zones'][1] + '/machineTypes/' +
context.properties['machine_type'],
            'metadata': {
                'items': [{
                    'key': 'user-data',
                    'value': context.properties['ignition']
                }]
            },
```

```
            'networkInterfaces': [{
                'subnetwork': context.properties['control_subnet']
            }],
            'serviceAccounts': [{
                'email': context.properties['service_account_email'],
                'scopes': ['https://www.googleapis.com/auth/cloud-platform']
            }],
            'tags': {
                'items': [
                    context.properties['infra_id'] + '-master',
                ]
            },
            'zone': context.properties['zones'][1]
        }
    }, {
        'name': context.properties['infra_id'] + '-master-2',
        'type': 'compute.v1.instance',
        'properties': {
            'disks': [{
                'autoDelete': True,
                'boot': True,
                'initializeParams': {
                    'diskSizeGb': context.properties['root_volume_size'],
                    'diskType': 'zones/' + context.properties['zones'][2] + '/diskTypes/pd-ssd',
                    'sourceImage': context.properties['image']
                }
            }],
            'machineType': 'zones/' + context.properties['zones'][2] + '/machineTypes/' +
context.properties['machine_type'],
            'metadata': {
                'items': [{
                    'key': 'user-data',
                    'value': context.properties['ignition']
                }]
            },
            'networkInterfaces': [{
                'subnetwork': context.properties['control_subnet']
            }],
            'serviceAccounts': [{
                'email': context.properties['service_account_email'],
                'scopes': ['https://www.googleapis.com/auth/cloud-platform']
            }],
            'tags': {
                'items': [
                    context.properties['infra_id'] + '-master',
                ]
            },
            'zone': context.properties['zones'][2]
        }
    }]

    return {'resources': resources}
```

4.10.16. 等待 bootstrap 完成并删除 GCP 中的 bootstrap 资源

在 Google Cloud Platform (GCP) 中创建所有所需的基础架构后，请等待您通过安装程序生成的 Ignition 配置文件所置备的机器上完成 bootstrap 过程。

先决条件

- 配置 GCP 帐户。
- 为集群生成 Ignition 配置文件。
- 在 GCP 中创建和配置 VPC 及相关子网。
- 在 GCP 中创建和配置联网及负载均衡器。
- 创建 control plane 和计算角色。
- 创建 bootstrap 机器。
- 创建 control plane 机器。

流程

1. 更改到包含安装程序的目录，再运行以下命令：

```
$ ./openshift-install wait-for bootstrap-complete --dir <installation_directory> \ 1
    --log-level info 2
```

1. 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。
2. 要查看不同的安装详情，请指定 **warn**、**debug** 或 **error**，而不要指定 **info**。

如果命令退出且不显示 **FATAL** 警告，则代表您的 control plane 已被初始化。

2. 删除 bootstrap 资源：

```
$ gcloud compute backend-services remove-backend ${INFRA_ID}-api-internal-backend-
service --region=${REGION} --instance-group=${INFRA_ID}-bootstrap-instance-group --
instance-group-zone=${ZONE_0}
$ gsutil rm gs://${INFRA_ID}-bootstrap-ignition/bootstrap.ign
$ gsutil rb gs://${INFRA_ID}-bootstrap-ignition
$ gcloud deployment-manager deployments delete ${INFRA_ID}-bootstrap
```

4.10.17. 在 GCP 中创建额外的 worker 机器

您可以通过分散启动各个实例或利用集群外自动化流程（如自动缩放组），在 Google Cloud Platform (GCP) 中为您的集群创建 worker 机器。您还可以利用 OpenShift Container Platform 中的内置集群扩展机制和机器 API。

在本例中，您要使用 Deployment Manager 模板来手动启动一个实例。通过在文件中添加类型为 **06_worker.py** 的其他资源，即可启动其他实例。

注意

如果不使用提供的 Deployment Manager 模板来创建 worker 机器，您必须检查提供的信息并手动创建基础架构。如果集群没有正确初始化，您可能需要联系红帽支持并提供您的安装日志。

先决条件

- 配置 GCP 帐户。
- 为集群生成 Ignition 配置文件。
- 在 GCP 中创建和配置 VPC 及相关子网。
- 在 GCP 中创建和配置联网及负载均衡器。
- 创建 control plane 和计算角色。
- 创建 bootstrap 机器。
- 创建 control plane 机器。

流程

1. 复制本主题的 **worker 机器的 Deployment Manager 模板**部分中的模板，并将它以 **06_worker.py** 形式保存到计算机中。此模板描述了集群所需的 worker 机器。
2. 导出资源定义使用的变量。
   a. 导出托管计算机器的子网：

```
$ export COMPUTE_SUBNET=(`gcloud compute networks subnets describe
${HOST_PROJECT_COMPUTE_SUBNET} --region=${REGION} --project
${HOST_PROJECT} --account ${HOST_PROJECT_ACCOUNT} --format json | jq -r
.selfLink`)
```

   b. 为您的服务帐户导出电子邮件地址：

```
$ export WORKER_SERVICE_ACCOUNT=(`gcloud iam service-accounts list --filter
"email~^${INFRA_ID}-w@${PROJECT_NAME}." --format json | jq -r '.[0].email'`)
```

   c. 导出计算机器 Ignition 配置文件的位置：

```
$ export WORKER_IGNITION=`cat <installation_directory>/worker.ign`
```

3. 创建 **06_worker.yaml** 资源定义文件：

```
$ cat <<EOF >06_worker.yaml
imports:
- path: 06_worker.py

resources:
- name: 'worker-0' (1)
  type: 06_worker.py
  properties:
```

```
    infra_id: '${INFRA_ID}' (2)
    zone: '${ZONE_0}' (3)
    compute_subnet: '${COMPUTE_SUBNET}' (4)
    image: '${CLUSTER_IMAGE}' (5)
    machine_type: 'n1-standard-4' (6)
    root_volume_size: '128'
    service_account_email: '${WORKER_SERVICE_ACCOUNT}' (7)
    ignition: '${WORKER_IGNITION}' (8)
- name: 'worker-1'
  type: 06_worker.py
  properties:
    infra_id: '${INFRA_ID}' (9)
    zone: '${ZONE_1}' (10)
    compute_subnet: '${COMPUTE_SUBNET}' (11)
    image: '${CLUSTER_IMAGE}' (12)
    machine_type: 'n1-standard-4' (13)
    root_volume_size: '128'
    service_account_email: '${WORKER_SERVICE_ACCOUNT}' (14)
    ignition: '${WORKER_IGNITION}' (15)
EOF
```

(1) **name** 是 worker 机器的名称，如 **worker-0**。

(2) (9) **infra_id** 是来自于提取步骤的 **INFRA_ID** 基础架构名称。

(3) (10) **zone** 是 worker 机器要部署到的区域，如 **us-central1-a**。

(4) (11) **compute_subnet** 是计算子网的 **selfLink**。

(5) (12) **image** 是 RHCOS 镜像的 **selfLink** URL。

(6) (13) **machine_type** 是实例的机器类型，如 **n1-standard-4**。

(7) (14) **service_account_email** 是创建的 worker 服务帐户的电子邮件地址。

(8) (15) **ignition** 是 **worker.ign** 文件的内容。

4. 可选：如果要启动更多实例，请在 **06_worker.yaml** 资源定义文件中包含类型为 **06_worker.py** 的其他资源。

5. 使用 **gcloud** CLI 创建部署：

```
$ gcloud deployment-manager deployments create ${INFRA_ID}-worker --config
06_worker.yaml
```

4.10.17.1. worker 机器的 Deloyment Manager 模板

您可以使用以下 Deployment Manager 模板来部署 OpenShift Container Platform 集群所需的 worker 机器：

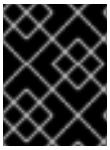
例 4.18. 06_worker.py Deployment Manager 模板

```
def GenerateConfig(context):
```

```
    resources = [{
        'name': context.properties['infra_id'] + '-' + context.env['name'],
        'type': 'compute.v1.instance',
        'properties': {
            'disks': [{
                'autoDelete': True,
                'boot': True,
                'initializeParams': {
                    'diskSizeGb': context.properties['root_volume_size'],
                    'sourceImage': context.properties['image']
                }
            }],
            'machineType': 'zones/' + context.properties['zone'] + '/machineTypes/' +
context.properties['machine_type'],
            'metadata': {
                'items': [{
                    'key': 'user-data',
                    'value': context.properties['ignition']
                }]
            },
            'networkInterfaces': [{
                'subnetwork': context.properties['compute_subnet']
            }],
            'serviceAccounts': [{
                'email': context.properties['service_account_email'],
                'scopes': ['https://www.googleapis.com/auth/cloud-platform']
            }],
            'tags': {
                'items': [
                    context.properties['infra_id'] + '-worker',
                ]
            },
            'zone': context.properties['zone']
        }
    }]

    return {'resources': resources}
```

4.10.18. 通过下载二进制文件安装 OpenShift CLI

您需要安装 CLI（**oc**） 来使用命令行界面与 OpenShift Container Platform 进行交互。您可在 Linux 、Windows 或 macOS 上安装 **oc**。



重要

如果安装了旧版本的 **oc**，则无法使用 OpenShift Container Platform 4.6 中的所有命令。下载并安装新版本的 **oc**。

4.10.18.1. 在 Linux 上安装 OpenShift CLI

您可以按照以下流程在 Linux 上安装 OpenShift CLI（**oc**）二进制文件。

流程

1. 进入到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 在 **Version** 下拉菜单中选择相应的版本。
3. 单击 **OpenShift v4.6 Linux 客户端** 条目旁边的 **Download Now**，再保存文件。
4. 解包存档：

```
$ tar xvzf <file>
```

5. 把 **oc** 二进制代码放到 **PATH** 中的目录中。
   执行以下命令可以查看当前的 **PATH** 设置：

```
$ echo $PATH
```

安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
$ oc <command>
```

4.10.18.2. 在 Windows 上安装 OpenShift CLI

您可以按照以下流程在 Windows 上安装 OpenShift CLI（**oc**）二进制代码。

流程

1. 进入到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 在 **Version** 下拉菜单中选择相应的版本。
3. 单击 **OpenShift v4.6 Windows 客户端**条目旁边的 **Download Now**，再保存文件。
4. 使用 ZIP 程序解压存档。
5. 把 **oc** 二进制代码放到 **PATH** 中的目录中。
   要查看您的 **PATH**，请打开命令提示窗口并执行以下命令：

```
C:\> path
```

安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
C:\> oc <command>
```

4.10.18.3. 在 macOS 上安装 OpenShift CLI

您可以按照以下流程在 macOS 上安装 OpenShift CLI（**oc**）二进制代码。

流程

1. 进入到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 在 **Version** 下拉菜单中选择相应的版本。

3. 单击 **OpenShift v4.6 MacOSX 客户端**条目旁边的 **Download Now**，再保存文件。
4. 解包和解压存档。
5. 将 **oc** 二进制文件移到 PATH 的目录中。
   要查看您的 **PATH**，打开一个终端窗口并执行以下命令：

```
$ echo $PATH
```

安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
$ oc <command>
```

4.10.19. 使用 CLI 登录到集群

您可以通过导出集群 **kubeconfig** 文件，以默认系统用户身份登录集群。**kubeconfig** 文件包含关于集群的信息，供 CLI 用于将客户端连接到正确集群和 API 服务器。该文件特只适用于一个特定的集群，在 OpenShift Container Platform 安装过程中创建。

先决条件

- 已部署了 OpenShift Container Platform 集群。
- 已安装 **oc** CLI。

流程

1. 导出 **kubeadmin** 凭证：

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig
```
(1)

(1) 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

2. 使用导出的配置，验证能否成功运行 **oc** 命令：

```
$ oc whoami
```

输出示例

```
system:admin
```

4.10.20. 批准机器的证书签名请求

将机器添加到集群时，会为您添加的每台机器生成两个待处理证书签名请求（CSR）。您必须确认这些 CSR 已获得批准，或根据需要自行批准。客户端请求必须首先被批准，然后是服务器请求。

先决条件

- 您已将机器添加到集群中。

流程

1. 确认集群可以识别这些机器：

```
$ oc get nodes
```

输出示例

```
NAME     STATUS   ROLES   AGE  VERSION
master-0 Ready    master  63m  v1.19.0
master-1 Ready    master  63m  v1.19.0
master-2 Ready    master  64m  v1.19.0
```

输出将列出您创建的所有机器。

注意

在一些 CSR 被批准前，以上输出可能不包括计算节点（也称为 worker 节点）。

2. 检查待处理的 CSR，并确保可以看到添加到集群中的每台机器都有 **Pending** 或 **Approved** 状态的客户端请求：

```
$ oc get csr
```

输出示例

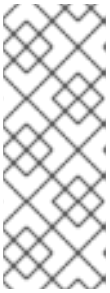
```
NAME        AGE    REQUESTOR                                                    CONDITION
csr-8b2br   15m    system:serviceaccount:openshift-machine-config-operator:node-
bootstrapper   Pending
csr-8vnps   15m    system:serviceaccount:openshift-machine-config-operator:node-
bootstrapper   Pending
...
```

在本例中，两台机器加入了集群。您可能会在列表中看到更多已批准的 CSR。

3. 如果 CSR 没有获得批准，请在所添加机器的所有待处理 CSR 都处于 **Pending** 状态后，为您的集群机器批准这些 CSR：

注意

由于 CSR 会自动轮转，因此请在将机器添加到集群后一小时内批准您的 CSR。如果没有在一小时内批准，证书将会轮转，每个节点将会存在多个证书。您必须批准所有这些证书。批准客户端 CSR 后， Kubelet 为服务证书创建辅助 CSR，这需要手动批准。然后，如果 Kubelet 请求具有相同参数的新证书，则 **machine-approver** 会自动批准后续服务证书续订请求。



注意

对于在未启用机器 API 的平台中运行的集群，如裸机和其他用户置备的基础架构，必须采用一种方法自动批准 kubelet 提供证书请求（CSR）。如果没有批准请求，则 **oc exec**、**oc rsh** 和 **oc logs** 命令将无法成功，因为 API 服务器连接到 kubelet 时需要服务证书。与 Kubelet 端点联系的任何操作都需要此证书批准。这个方法必须监视新的 CSR，确认 CSR 由 **system:node** 或 **system:admin** 组中的 **node-bootstrapper** 服务帐户提交，并确认节点的身份。

- 若要单独批准，请对每个有效的 CSR 运行以下命令：

```
$ oc adm certificate approve <csr_name> 1
```

1 **<csr_name>** 是当前 CSR 列表中 CSR 的名称。

- 要批准所有待处理的 CSR，请运行以下命令：

```
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}{{"\n"}}
{{end}}{{end}}' | xargs --no-run-if-empty oc adm certificate approve
```

注意

在有些 CSR 被批准前，一些 Operator 可能无法使用。

4. 现在，您的客户端请求已被批准，您必须查看添加到集群中的每台机器的服务器请求：

```
$ oc get csr
```

输出示例

```
NAME        AGE     REQUESTOR                                                   CONDITION
csr-bfd72   5m26s   system:node:ip-10-0-50-126.us-east-2.compute.internal
Pending
csr-c57lv   5m26s   system:node:ip-10-0-95-157.us-east-2.compute.internal
Pending
...
```

5. 如果剩余的 CSR 没有被批准，且处于 **Pending** 状态，请批准集群机器的 CSR：

- 若要单独批准，请对每个有效的 CSR 运行以下命令：

```
$ oc adm certificate approve <csr_name> 1
```

1 **<csr_name>** 是当前 CSR 列表中 CSR 的名称。

- 要批准所有待处理的 CSR，请运行以下命令：

```
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}{{"\n"}}
{{end}}{{end}}' | xargs oc adm certificate approve
```

6. 批准所有客户端和服务器 CSR 后，器将处于 **Ready** 状态。运行以下命令验证：

```
$ oc get nodes
```

输出示例

```
NAME      STATUS    ROLES   AGE  VERSION
master-0  Ready     master  73m  v1.20.0
master-1  Ready     master  73m  v1.20.0
```

```
master-2  Ready    master  74m  v1.20.0
worker-0  Ready    worker  11m  v1.20.0
worker-1  Ready    worker  11m  v1.20.0
```

注意

批准服务器 CSR 后可能需要几分钟时间让机器转换为 **Ready** 状态。

其他信息

- 如需有关 CSR 的更多信息，请参阅证书签名请求。

4.10.21. 添加 Ingress DNS 记录

在创建 Kubernetes 清单并生成 Ignition 配置时会删除 DNS 区配置。您必须手动创建指向入口负载均衡器的 DNS 记录。您可以创建通配符 ***.apps.{baseDomain}.** 或具体的记录。您可以根据自己的要求使用 A、CNAME 和其他记录。

先决条件

- 配置 GCP 帐户。
- 在创建 Kubernetes 清单并生成 Ignition 配置时删除 DNS 区配置。
- 在 GCP 中创建和配置 VPC 及相关子网。
- 在 GCP 中创建和配置联网及负载均衡器。
- 创建 control plane 和计算角色。
- 创建 bootstrap 机器。
- 创建 control plane 机器。
- 创建 worker 机器。

流程

1. 等待入口路由器创建负载均衡器并填充 **EXTERNAL-IP** 字段：

```
$ oc -n openshift-ingress get service router-default
```

输出示例

```
NAME             TYPE           CLUSTER-IP      EXTERNAL-IP     PORT(S)                      AGE
router-default   LoadBalancer   172.30.18.154   35.233.157.184
80:32288/TCP,443:31215/TCP   98
```

2. 在您的区中添加 A 记录：
   - 使用 A 记录：
     i. 为路由器 IP 地址导出变量：

```
$ export ROUTER_IP=`oc -n openshift-ingress get service router-default --no-
headers | awk '{print $4}'`
```

ii. 在专用区中添加 A 记录：

```
$ if [ -f transaction.yaml ]; then rm transaction.yaml; fi
$ gcloud dns record-sets transaction start --zone ${INFRA_ID}-private-zone --project
${HOST_PROJECT} --account ${HOST_PROJECT_ACCOUNT}
$ gcloud dns record-sets transaction add ${ROUTER_IP} --name
\*.apps.${CLUSTER_NAME}.${BASE_DOMAIN}. --ttl 300 --type A --zone
${INFRA_ID}-private-zone --project ${HOST_PROJECT} --account
${HOST_PROJECT_ACCOUNT}
$ gcloud dns record-sets transaction execute --zone ${INFRA_ID}-private-zone --
project ${HOST_PROJECT} --account ${HOST_PROJECT_ACCOUNT}
```

iii. 对于外部集群，还要将 A 记录添加到公共区：

```
$ if [ -f transaction.yaml ]; then rm transaction.yaml; fi
$ gcloud dns record-sets transaction start --zone ${BASE_DOMAIN_ZONE_NAME} -
-project ${HOST_PROJECT} --account ${HOST_PROJECT_ACCOUNT}
$ gcloud dns record-sets transaction add ${ROUTER_IP} --name
\*.apps.${CLUSTER_NAME}.${BASE_DOMAIN}. --ttl 300 --type A --zone
${BASE_DOMAIN_ZONE_NAME} --project ${HOST_PROJECT} --account
${HOST_PROJECT_ACCOUNT}
$ gcloud dns record-sets transaction execute --zone
${BASE_DOMAIN_ZONE_NAME} --project ${HOST_PROJECT} --account
${HOST_PROJECT_ACCOUNT}
```

- 如果需要添加特定域而不使用通配符，可以为集群的每个当前路由创建条目：

```
$ oc get --all-namespaces -o jsonpath='{range .items[*]}{range .status.ingress[*]}{.host}
{"\n"}{end}{end}' routes
```

输出示例

```
oauth-openshift.apps.your.cluster.domain.example.com
console-openshift-console.apps.your.cluster.domain.example.com
downloads-openshift-console.apps.your.cluster.domain.example.com
alertmanager-main-openshift-monitoring.apps.your.cluster.domain.example.com
grafana-openshift-monitoring.apps.your.cluster.domain.example.com
prometheus-k8s-openshift-monitoring.apps.your.cluster.domain.example.com
```

4.10.22. 添加入口防火墙规则

集群需要多个防火墙规则。如果不使用共享 VPC，则由 ingress 控制器通过 GCP 云供应商创建这些规则。使用共享 VPC 时，现在可在集群请求访问时为所有服务创建集群范围的防火墙规则，或者根据事件创建每个规则。在集群请求访问时，通过创建每个规则，您需要知道具体的防火墙规则。通过创建集群范围的防火墙规则，您可以在多个集群中应用相同的规则。

如果您选择基于事件创建每个规则，必须在置备集群后创建防火墙规则，并在控制台通知您缺少规则时在集群生命周期中创建防火墙规则。此时会显示类似以下事件的事件，您必须添加所需防火墙规则：

```
$ oc get events -n openshift-ingress --field-selector="reason=LoadBalancerManualChange"
```

输出示例

```
Firewall change required by security admin: `gcloud compute firewall-rules create k8s-fw-a26e631036a3f46cba28f8df67266d55 --network example-network --description "{\"kubernetes.io/service-name\":\"openshift-ingress/router-default\", \"kubernetes.io/service-ip\":\"35.237.236.234\"}\" --allow tcp:443,tcp:80 --source-ranges 0.0.0.0/0 --target-tags exampl-fqzq7-master,exampl-fqzq7-worker --project example-project`
```

如果您在创建这些基于规则的事件时遇到问题，可以在集群运行时配置集群范围的防火墙规则。

4.10.22.1. 在 GCP 中为共享 VPC 创建集群范围的防火墙规则

您可以创建集群范围的防火墙规则，以允许 OpenShift Container Platform 集群所需的访问。

警告

如果您没有选择基于集群事件创建防火墙规则，您必须创建集群范围的防火墙规则。

先决条件

- 您导出了部署集群的 Deployment Manager 模板所需的变量。
- 您在集群所需的 GCP 中创建了网络和负载均衡组件。

流程

1. 添加单个防火墙规则，允许 Google Cloud Engine 健康检查访问所有服务。此规则可让入口负载均衡器决定其实例的健康状况。

```
$ gcloud compute firewall-rules create --allow='tcp:30000-32767,udp:30000-32767' --network="${CLUSTER_NETWORK}" --source-ranges='130.211.0.0/22,35.191.0.0/16,209.85.152.0/22,209.85.204.0/22' --target-tags="${INFRA_ID}-master,${INFRA_ID}-worker" ${INFRA_ID}-ingress-hc --account=${HOST_PROJECT_ACCOUNT} --project=${HOST_PROJECT}
```

2. 添加一个防火墙规则来允许访问所有群集服务：

   - 对于外部集群：

```
$ gcloud compute firewall-rules create --allow='tcp:80,tcp:443' --network="${CLUSTER_NETWORK}" --source-ranges="0.0.0.0/0" --target-tags="${INFRA_ID}-master,${INFRA_ID}-worker" ${INFRA_ID}-ingress --account=${HOST_PROJECT_ACCOUNT} --project=${HOST_PROJECT}
```

   - 对于私有集群：

```
$ gcloud compute firewall-rules create --allow='tcp:80,tcp:443' --network="${CLUSTER_NETWORK}" --source-ranges=${NETWORK_CIDR} --target-tags="${INFRA_ID}-master,${INFRA_ID}-worker" ${INFRA_ID}-ingress --
```

```
account=${HOST_PROJECT_ACCOUNT} --project=${HOST_PROJECT}
```

因为这个规则只允许 TCP 端口 **80** 和 **443** 的流量，请确定您添加了所有服务使用的端口。

4.10.23. 在用户置备的基础架构上完成 GCP 安装

在 Google Cloud Platform (GCP) 用户置备的基础架构上启动 OpenShift Container Platform 安装后，您可以监控集群事件，直到集群就绪可用。

先决条件

- 在用户置备的 GCP 基础架构上为 OpenShift Container Platform 集群部署 bootstrap 机器。
- 安装 **oc** CLI 并登录。

流程

1. 完成集群安装：

```
$ ./openshift-install --dir <installation_directory> wait-for install-complete 1
```

输出示例

```
INFO Waiting up to 30m0s for the cluster to initialize...
```

1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

重要

- 安装程序生成的 Ignition 配置文件包含在 24 小时后过期的证书，然后在过期时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外情况是，您需要手动批准待处理的 **node-bootstrapper** 证书签名请求（CSR）来恢复 kubelet 证书。如需更多信息，请参阅*从过期的 control plane 证书中恢复*的文档。
- 建议您在生成 12 小时后使用 Ignition 配置文件，因为集群安装后 24 小时证书从 16 小时轮转至 22 小时。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中运行证书更新时避免安装失败。

2. 观察集群的运行状态。

   a. 运行以下命令来查看当前的集群版本和状态：

```
$ oc get clusterversion
```

   输出示例

```
NAME      VERSION   AVAILABLE   PROGRESSING   SINCE   STATUS
version             False       True          24m     Working towards 4.5.4: 99% complete
```

   b. 运行以下命令，查看 control plane 上由 Cluster Version Operator (CVO) 管理的 Operator：

```
$ oc get clusteroperators
```

输出示例

```
NAME                                       VERSION   AVAILABLE   PROGRESSING   DEGRADED
SINCE
authentication                             4.5.4     True        False         False      7m56s
cloud-credential                           4.5.4     True        False         False      31m
cluster-autoscaler                         4.5.4     True        False         False      16m
console                                    4.5.4     True        False         False      10m
csi-snapshot-controller                    4.5.4     True        False         False      16m
dns                                        4.5.4     True        False         False      22m
etcd                                       4.5.4     False       False         False      25s
image-registry                             4.5.4     True        False         False      16m
ingress                                    4.5.4     True        False         False      16m
insights                                   4.5.4     True        False         False      17m
kube-apiserver                             4.5.4     True        False         False      19m
kube-controller-manager                    4.5.4     True        False         False      20m
kube-scheduler                             4.5.4     True        False         False      20m
kube-storage-version-migrator              4.5.4     True        False         False      16m
machine-api                                4.5.4     True        False         False      22m
machine-config                             4.5.4     True        False         False      22m
marketplace                                4.5.4     True        False         False      16m
monitoring                                 4.5.4     True        False         False      10m
network                                    4.5.4     True        False         False      23m
node-tuning                                4.5.4     True        False         False      23m
openshift-apiserver                        4.5.4     True        False         False      17m
openshift-controller-manager               4.5.4     True        False         False      15m
openshift-samples                          4.5.4     True        False         False      16m
operator-lifecycle-manager                 4.5.4     True        False         False      22m
operator-lifecycle-manager-catalog         4.5.4     True        False         False      22m
operator-lifecycle-manager-packageserver   4.5.4     True        False         False      18m
service-ca                                 4.5.4     True        False         False      23m
service-catalog-apiserver                  4.5.4     True        False         False      23m
service-catalog-controller-manager         4.5.4     True        False         False      23m
storage                                    4.5.4     True        False         False      17m
```

c. 运行以下命令来查看您的集群 pod：

```
$ oc get pods --all-namespaces
```

输出示例

```
NAMESPACE                                  NAME
READY   STATUS    RESTARTS   AGE
kube-system                                etcd-member-ip-10-0-3-111.us-east-
2.compute.internal             1/1     Running   0          35m
kube-system                                etcd-member-ip-10-0-3-239.us-east-
2.compute.internal             1/1     Running   0          37m
kube-system                                etcd-member-ip-10-0-3-24.us-east-
2.compute.internal             1/1     Running   0          35m
openshift-apiserver-operator               openshift-apiserver-operator-6d6674f4f4-
h7t2t                          1/1     Running   1          37m
openshift-apiserver                        apiserver-fm48r
```

```
1/1     Running   0          30m
openshift-apiserver                                apiserver-fxkvv
1/1     Running   0          29m
openshift-apiserver                                apiserver-q85nm
1/1     Running   0          29m
...
openshift-service-ca-operator                      openshift-service-ca-operator-66ff6dc6cd-
9r257                      1/1     Running   0          37m
openshift-service-ca                               apiservice-cabundle-injector-695b6bcbc-cl5hm
1/1     Running   0          35m
openshift-service-ca                               configmap-cabundle-injector-8498544d7-
25qn6                      1/1     Running   0          35m
openshift-service-ca                               service-serving-cert-signer-6445fc9c6-wqdqn
1/1     Running   0          35m
openshift-service-catalog-apiserver-operator       openshift-service-catalog-apiserver-
operator-549f44668b-b5q2w      1/1     Running   0          32m
openshift-service-catalog-controller-manager-operator   openshift-service-catalog-
controller-manager-operator-b78cr2lnm   1/1     Running   0          31m
```

当前集群版本是 **AVAILABLE** 时表示安装完毕。

4.10.24. OpenShift Container Platform 的 Telemetry 访问

在 OpenShift Container Platform 4.6 中，默认运行的 Telemetry 服务提供有关集群健康状况和成功更新的指标，需要访问互联网。如果您的集群连接到互联网，Telemetry 会自动运行，而且集群会注册到 OpenShift Cluster Manager。

确认 OpenShift Cluster Manager 清单正确后，可以由 Telemetry 自动维护，也可以使用 OpenShift Cluster Manager 手动维护，使用订阅监控来跟踪帐户或多集群级别的 OpenShift Container Platform 订阅。

其他资源

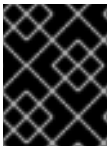
- 有关 Telemetry 服务的更多信息，请参阅关于远程健康监控。

4.10.25. 后续步骤

- 自定义集群。
- 如果需要，您可以选择不使用远程健康报告。

4.11. 在带有用户置备的受限网络中的 GCP 上安装集群

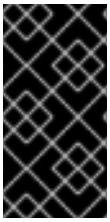
在 OpenShift Container Platform 版本 4.6 中，您可以使用您提供的基础架构和安装发行内容的内部镜像在 Google Cloud Platform（GCP）上安装集群。



重要

虽然您可以使用镜像安装发行内容安装 OpenShift Container Platform 集群，但您的集群仍需要访问互联网才能使用 GCP API。

此处概述了进行用户提供基础架构安装的步骤。提供的几个 Deployment Manager 模板可协助完成这些步骤，也可帮助您自行建模。您还可以自由选择通过其他方法创建所需的资源。



重要

进行用户置备的基础架构安装的步骤仅作为示例。使用您提供的基础架构安装集群需要了解云供应商和 OpenShift Container Platform 安装过程。提供的几个 Deployment Manager 模板可帮助完成这些步骤，或者帮助您自行建模。您也可以自由选择通过其他方法创建所需的资源；模板仅作示例之用。

4.11.1. 先决条件

- 在镜像主机上创建镜像 registry，并获取您的 OpenShift Container Platform 版本的 **imageContentSources** 数据。

  重要

  由于安装介质位于堡垒主机上，因此请使用该计算机完成所有安装步骤。

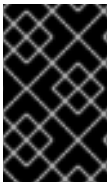
- 查看有关 OpenShift Container Platform 安装和更新流程的详细信息。
- 如果使用防火墙，则必须将其配置为允许集群需要访问的站点。虽然您可能需要授予更多站点的访问权限，但您必须授予对 ***.googleapis.com** 和 **accounts.google.com** 的访问权限。
- 如果不允许系统管理身份和访问管理（IAM），集群管理员可以 手动创建和维护 IAM 凭证。手动模式也可以用于云 IAM API 无法访问的环境中。

4.11.2. 关于在受限网络中安装

在 OpenShift Container Platform 4.6 中，可以执行不需要有效的互联网连接来获取软件组件的安装。受限网络安装可使用安装程序置备的基础架构或用户置备的基础架构完成，具体取决于您要安装集群的云平台。

如果选择在云平台中执行受限网络安装，仍然需要访问其云 API。有些云功能，比如 Amazon Web Service 的 Route 53 DNS 和 IAM 服务，需要访问互联网。根据您的网络，在裸机硬件或 VMware vSphere 上安装时可能需要较少的互联网访问。

要完成受限网络安装，您必须创建一个 registry，镜像 OpenShift Container Platform registry 的内容并包含其安装介质。您可以在堡垒主机上创建此镜像，该主机可同时访问互联网和您的封闭网络，也可以使用满足您的限制条件的其他方法。



重要

由于用户置备安装配置的复杂性，在尝试使用用户置备的基础架构受限网络安装前，请考虑完成标准用户置备的基础架构安装。通过完成此测试安装，您可以更轻松地隔离和排查您在受限网络中安装时可能出现的问题。

4.11.2.1. 其他限制

受限网络中的集群还有以下额外限制：

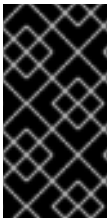
- **ClusterVersion** 状态包含一个 **Unable to retrieve available updates** 错误。
- 默认情况下，您无法使用 Developer Catalog 的内容，因为您无法访问所需的镜像流标签。

4.11.3. OpenShift Container Platform 的互联网访问

在 OpenShift Container Platform 4.6 中，您需要访问互联网来获得用来安装集群的镜像。

您必须具有以下互联网访问权限：

- 访问 OpenShift Cluster Manager 以下载安装程序并执行订阅管理。如果集群可以访问互联网，并且没有禁用 Telemetry，该服务会自动授权您的集群。
- 访问 Quay.io，以获取安装集群所需的软件包。
- 获取执行集群更新所需的软件包。



重要

如果您的集群无法直接访问互联网，则可以在置备的某些类基础架构上执行受限网络安装。在此过程中，您要下载所需的内容，并使用它在镜像 registry（mirror registry） 中填充安装集群并生成安装程序所需的软件包。对于某些安装类型，集群要安装到的环境不需要访问互联网。在更新集群之前，要更新 registry 镜像系统中的内容。

4.11.4. 配置 GCP 项目

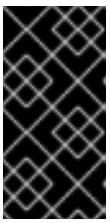
在安装 OpenShift Container Platform 之前，您必须配置 Google Cloud Platform (GCP) 项目来托管它。

4.11.4.1. 创建一个 GCP 项目

为了安装 OpenShift Container Platform，您必须在您的 Google Cloud Platform (GCP) 账户中创建一个项目来托管它。

流程

- 创建一个项目来托管您的 OpenShift Container Platform 集群。请参阅 GCP 文档中的创建和管理项目。



重要

如果您使用安装程序置备的基础架构，您的 GCP 项目必须使用 Premium Network Service Tier。使用安装程序安装的集群不支持 Standard Network Service Tier。安装程序为 **api-int.<cluster_name>.<base_domain>** URL 配置内部负载均衡；内部负载均衡需要 Premium Tier。

4.11.4.2. 在 GCP 中启用 API 服务

Google Cloud Platform (GCP) 项目需要访问一些 API 服务来完成 OpenShift Container Platform 安装。

先决条件

- 创建了用于托管集群的项目。

流程

- 在托管集群的项目中启用如下所需的 API 服务。请参阅 GCP 文档中的启用服务。

  表 4.45. 所需的 API 服务

API 服务	控制台服务名称
compute Engine API	**compute.googleapis.com**
Google Cloud API	**cloudapis.googleapis.com**
Cloud Resource Manager API	**cloudresourcemanager.googleapis.com**
Google DNS API	**dns.googleapis.com**
IAM Service Account Credentials API	**iamcredentials.googleapis.com**
Identity and Access Management (IAM) API	**iam.googleapis.com**
Service Management API	**servicemanagement.googleapis.com**
Service Usage API	**serviceusage.googleapis.com**
Google Cloud Storage JSON API	**storage-api.googleapis.com**
Cloud Storage	**storage-component.googleapis.com**

4.11.4.3. 为 GCP 配置 DNS

要安装 OpenShift Container Platform，您使用的 Google Cloud Platform (GCP) 帐户必须在托管 OpenShift Container Platform 集群的同一项目中有一个专用的公共托管区。此区域必须对域具有权威。DNS 服务为集群外部连接提供集群 DNS 解析和名称查询。

流程

1. 标识您的域或子域，以及注册商（registrar）。您可以转移现有的域和注册商，或通过 GCP 或其他来源获取新的域和注册商。

注意

如果您购买新域，则需要时间来传播相关的 DNS 更改。有关通过 Google 购买域的更多信息，请参阅 Google Domains。

2. 在 GCP 项目中为您的域或子域创建一个公共托管区。请参阅 GCP 文档中的创建公共区。使用合适的根域（如 **openshiftcorp.com**）或子域（如 **clusters.openshiftcorp.com**）。

3. 从托管区记录中提取新的权威名称服务器。请参阅 GCP 文档中的查找您的云 DNS 名称服务器。您通常有四个名称服务器。

4. 更新域所用名称服务器的注册商记录。例如，如果您将域注册到了 Google Domains，请参阅 Google Domains 帮助中的以下主题：如何切换到自定义名称服务器。

5. 如果您将根域迁移到 Google Cloud DNS，请迁移您的 DNS 记录。请参阅 GCP 文档中的 Migrating to Cloud DNS 。

6. 如果您使用子域，请按照您公司的流程将其委派记录添加到父域。这个过程可能包括对您公司的 IT 部门或控制您公司的根域和 DNS 服务的部门提出请求。

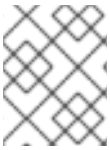
4.11.4.4. GCP 帐户限值

OpenShift Container Platform 集群使用诸多 Google Cloud Platform (GCP) 组件，默认的配额不会影响您安装默认 OpenShift Container Platform 集群的能力。

默认集群中包含三台计算机器和三台 control plane 机器，使用了以下资源。请注意，一些资源只在 bootstrap 过程中需要，会在集群部署后删除。

表 4.46. 默认集群中使用的 GCP 资源

服务	组件	位置	所需的资源总数	bootstrap 后删除的资源
服务帐户	IAM	全局	5	0
防火墙规则	网络	全局	11	1
转发规则	Compute	全局	2	0
健康检查	Compute	全局	2	0
镜像	Compute	全局	1	0
网络	网络	全局	1	0
路由器	网络	全局	1	0
Routes	网络	全局	2	0
子网	Compute	全局	2	0
目标池	网络	全局	2	0



注意

如果在安装过程中存在任何配额不足的情况，安装程序会显示一个错误信息，包括超过哪个配额，以及相关的区域。

请考虑您的集群的实际大小、预定的集群增长以及来自与您的帐户关联的其它集群的使用情况。CPU 、静态 IP 地址和持久性磁盘 SSD (storage) 配额是最可能不足的。

如果您计划在以下区域之一部署集群，您将超过最大存储配额，并可能会超过 CPU 配额限制：

- **asia-east2**

- **asia-northeast2**
- **asia-south1**
- **domain-southeast1**
- **europe-north1**
- **europe-west2**
- **europe-west3**
- **europe-west6**
- **northamerica-northeast1**
- **southamerica-east1**
- **us-west2**

您可以从 GCP 控制台增加资源配额，但可能需要提交一个支持问题单。务必提前规划集群大小，以便在安装 OpenShift Container Platform 集群前有足够的时间来等待支持问题单被处理。

4.11.4.5. 在 GCP 中创建服务帐户

OpenShift Container Platform 需要一个 Google Cloud Platform (GCP) 服务帐户，用于访问 Google API 中数据的验证和授权。如果您没有包含项目中所需角色的现有 IAM 服务帐户，您必须创建一个。

先决条件

- 创建了用于托管集群的项目。

流程

1. 在用于托管 OpenShift Container Platform 集群的项目中，创建一个新服务帐户。请参阅 GCP 文档中的创建服务账户。
2. 为服务帐户授予适当的权限。您可以逐一授予权限，也可以为其分配 **Owner** 角色。请参阅将角色授予特定资源的服务帐户。

注意

获得所需权限最简单的方法是把服务帐户设置为项目的拥有者（owner），这意味着该服务帐户对项目有完全的控制权。您必须考虑这样设定所带来的的风险是否可以接受。

3. 以 JSON 格式创建服务帐户密钥。请参阅 GCP 文档中的创建服务帐户密钥。
创建集群时需要该服务帐户密钥。

4.11.4.5.1. 所需的 GCP 权限

如果将 **Owner** 角色附加到您创建的服务帐户，您向该服务帐户授予所有的权限，包括安装 OpenShift Container Platform 所需的权限。要部署 OpenShift Container Platform 集群，服务帐户需要以下权限：
如果您将集群部署到现有的 VPC 中，则服务帐户不需要某些网络权限，如下表所示：

安装程序所需的角色

- Compute Admin
- Security Admin
- Service Account Admin
- Service Account User
- Storage Admin

安装过程中创建网络资源所需的角色

- DNS Administrator

用户置备的 GCP 基础架构所需的角色

- Deployment Manager Editor
- Service Account Key Admin

可选角色

若要使集群为其 Operator 创建新的有限凭证，请添加以下角色：

- Service Account Key Admin

以下角色将应用到 control plane 或计算机器使用的服务帐户：

表 4.47. GCP 服务帐户权限

帐户	角色
Control Plane	**roles/compute.instanceAdmin**
	roles/compute.networkAdmin
	roles/compute.securityAdmin
	roles/storage.admin
	roles/iam.serviceAccountUser
Compute	**roles/compute.viewer**
	roles/storage.admin

4.11.4.6. 支持的 GCP 区域

您可以将 OpenShift Container Platform 集群部署到下列 Google Cloud Platform (GCP) 区域：

- **asia-east1** (Changhua County, Taiwan)

- **asia-east2** (Hong Kong)
- **asia-northeast1** (Tokyo, Japan)
- **asia-northeast2** (Osaka, Japan)
- **asia-northeast3** (Seoul, South Korea)
- **asia-south1** (Mumbai, India)
- **asia-southeast1** (Jurong West, Singapore)
- **asia-southeast2** (Jakarta, India)
- **australia-southeast1** (Sydney, Australia)
- **europe-north1** (Hamina, Finland)
- **europe-west1** (St. Ghislain, Belgium)
- **europe-west2** (London, England, UK)
- **europe-west3** (Frankfurt, Germany)
- **europe-west4** (Eemshaven, Netherlands)
- **europe-west6** (Zürich, Switzerland)
- **northamerica-northeast1** (Montréal, Québec, Canada)
- **southamerica-east1** (São Paulo, Brazil)
- **us-central1** (Council Bluffs, Iowa, USA)
- **us-east1** (Moncks Corner, South Carolina, USA)
- **us-east4** (Ashburn, Northern Virginia, USA)
- **us-west1** (The Dalles, Oregon, USA)
- **us-west2** (Los Angeles, California, USA)
- **us-west3** (Salt Lake City, Utah, USA)
- **us-west4** (Las Vegas, Nevada, USA)

4.11.4.7. 为 GCP 安装和配置 CLI 工具

要使用用户置备的基础架构在 Google Cloud Platform (GCP) 上安装 OpenShift Container Platform，您必须为 GCP 安装和配置 CLI 工具。

先决条件

- 创建了用于托管集群的项目。
- 您创建了服务帐户，并授予其所需的权限。

流程

流程

1. 在 **$PATH** 中安装以下二进制文件：
   - **gcloud**
   - **gsutil**

   请参阅 GCP 文档中的安装最新 Cloud SDK 版本。
2. 使用 **gcloud** 工具及您配置的服务帐户进行身份验证。
   请参阅 GCP 文档中的使用服务帐户授权。

4.11.5. 为 GCP 创建安装文件

要使用用户置备的基础架构在 Google Cloud Platform (GCP) 上安装 OpenShift Container Platform，您必须生成并修改安装程序部署集群所需的文件，以便集群只创建要使用的机器。您要生成并自定义 **install-config.yaml** 文件、Kubernetes 清单和 Ignition 配置文件。您也可以选择在安装准备阶段首先设置独立的 **var** 分区。

4.11.5.1. 可选：创建独立 /var 分区

建议安装程序将 OpenShift Container Platform 的磁盘分区保留给安装程序。然而，在有些情况下您可能需要在文件系统的一部分中创建独立分区。

OpenShift Container Platform 支持添加单个分区来将存储附加到 **/var** 分区或 **/var** 的子目录。例如：

- **/var/lib/containers**：保存镜像相关的内容，随着更多镜像和容器添加到系统中，它所占用的存储会增加。
- **/var/lib/etcd**：保存您可能希望保持独立的数据，比如 etcd 存储的性能优化。
- **/var**：保存您希望独立保留的数据，用于特定目的（如审计）。

单独存储 **/var** 目录的内容可方便地根据需要对区域扩展存储，并可以在以后重新安装 OpenShift Container Platform 时保持该数据地完整。使用这个方法，您不必再次拉取所有容器，在更新系统时也无法复制大量日志文件。

因为 **/var** 在进行一个全新的 Red Hat Enterprise Linux CoreOS（RHCOS）安装前必需存在，所以这个流程会在 OpenShift Container Platform 安装过程的 **openshift-install** 准备阶段插入的机器配置来设置独立的 **/var** 分区。

重要

如果按照以下步骤在此流程中创建独立 **/var** 分区，则不需要再次创建 Kubernetes 清单和 Ignition 配置文件，如本节所述。

流程

1. 创建存放 OpenShift Container Platform 安装文件的目录：

```
$ mkdir $HOME/clusterconfig
```

2. 运行 **openshift-install** 在 **manifest** 和 **openshift** 子目录中创建一组文件。在出现提示时回答系统问题：

```
$ openshift-install create manifests --dir $HOME/clusterconfig
```

输出示例

```
? SSH Public Key ...
INFO Credentials loaded from the "myprofile" profile in file "/home/myuser/.aws/credentials"
INFO Consuming Install Config from target directory
INFO Manifests created in: $HOME/clusterconfig/manifests and
$HOME/clusterconfig/openshift
```

3. 可选：确认安装程序在 **clusterconfig/openshift** 目录中创建了清单：

```
$ ls $HOME/clusterconfig/openshift/
```

输出示例

```
99_kubeadmin-password-secret.yaml
99_openshift-cluster-api_master-machines-0.yaml
99_openshift-cluster-api_master-machines-1.yaml
99_openshift-cluster-api_master-machines-2.yaml
...
```

4. 创建 **MachineConfig** 对象并将其添加到 **openshift** 目录中的一个文件中。例如，把文件命名为 **98-var-partition.yaml**，将磁盘设备名称改为 **worker** 系统中存储设备的名称，并根据情况设置存储大小。这个示例将 **/var** 目录放在一个单独的分区中：

```
apiVersion: machineconfiguration.openshift.io/v1
kind: MachineConfig
metadata:
  labels:
    machineconfiguration.openshift.io/role: worker
  name: 98-var-partition
spec:
  config:
    ignition:
      version: 3.1.0
    storage:
      disks:
      - device: /dev/<device_name> 1
        partitions:
        - label: var
          startMiB: <partition_start_offset> 2
          sizeMiB: <partition_size> 3
      filesystems:
        - device: /dev/disk/by-partlabel/var
          path: /var
          format: xfs
    systemd:
      units:
        - name: var.mount 4
          enabled: true
          contents: |
            [Unit]
```

```
Before=local-fs.target
[Mount]
What=/dev/disk/by-partlabel/var
Where=/var
Options=defaults,prjquota 5
[Install]
WantedBy=local-fs.target
```

1. 要分区的磁盘的存储设备名称。
2. 当在引导磁盘中添加数据分区时，推荐最少使用 25000 MiB（Mebibytes）。root 文件系统会自动重新定义大小使其占据所有可用空间（最多到指定的偏移值）。如果没有指定值，或者指定的值小于推荐的最小值，则生成的 root 文件系统会太小，而在以后进行的 RHCOS 重新安装可能会覆盖数据分区的开始部分。
3. 数据分区的大小（以兆字节为单位）。
4. 挂载单元的名称必须与 **Where=** 指令中指定的目录匹配。例如，对于挂载在 **/var/lib/containers** 上的文件系统，该单元必须命名为 **var-lib-containers.mount**。
5. 对于用于容器存储的文件系统，必须启用 **prjquota** 挂载选项。

注意

在创建独立 **/var** 分区时，如果不同的实例类型没有相同的设备名称，则无法将不同的实例类型用于 worker 节点。

5. 再次运行 **openshift-install**，从 **manifest** 和 **openshift** 子目录中的一组文件创建 Ignition 配置：

```
$ openshift-install create ignition-configs --dir $HOME/clusterconfig
$ ls $HOME/clusterconfig/
auth  bootstrap.ign  master.ign  metadata.json  worker.ign
```

现在，您可以使用 Ignition 配置文件作为安装程序的输入来安装 Red Hat Enterprise Linux CoreOS（RHCOS）系统。

4.11.5.2. 创建安装配置文件

您可以自定义在 Google Cloud Platform (GCP) 上安装的 OpenShift Container Platform 集群。

先决条件

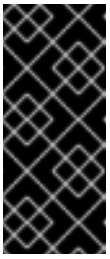
- 获取 OpenShift Container Platform 安装程序以及集群的 pull secret。对于受限网络安装，这些文件位于您的堡垒主机上。
- 具有创建镜像容器镜像仓库（registry）时生成的 **imageContentSources** 值。
- 获取您的镜像 registry 的证书内容。

流程

1. 创建 **install-config.yaml** 文件。
   a. 更改到包含安装程序的目录，再运行以下命令：

```
$ ./openshift-install create install-config --dir <installation_directory> 1
```

1 对于 **<installation_directory>**，请指定用于保存安装程序所创建的文件的目录名称。

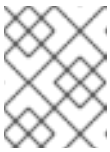


重要

指定一个空目录。一些安装信息，如 bootstrap X.509 证书，有较短的过期间隔，因此不要重复使用安装目录。如果要重复使用另一个集群安装中的个别文件，可以将其复制到您的目录中。但是，一些安装数据的文件名可能会在发行版本之间有所改变。从 OpenShift Container Platform 老版本中复制安装文件时要格外小心。

b. 在提示符处，提供您的云的配置详情：

i. 可选：选择用来访问集群机器的 SSH 密钥。



注意

对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 **ssh-agent** 进程使用的 SSH 密钥。

ii. 选择 **gcp** 作为目标平台。

iii. 如果您没有为计算机上的 GCP 帐户配置服务帐户密钥，则必须从 GCP 获取，并粘贴文件的内容或输入文件的绝对路径。

iv. 选择要在其中置备集群的项目 ID。默认值由您配置的服务帐户指定。

v. 选择要在其中部署集群的区域。

vi. 选择集群要部署到的基域。基域与您为集群创建的公共 DNS 区对应。

vii. 为集群输入一个描述性名称。

viii. 粘贴 Red Hat OpenShift Cluster Manager 中的 pull secret 。

2. 编辑 **install-config.yaml** 文件，以提供在受限网络中安装所需的其他信息。

a. 更新 **pullSecret** 值，使其包含 registry 的身份验证信息：

```
pullSecret: '{"auths":{"<mirror_host_name>:5000": {"auth": "<credentials>","email":
"you@example.com"}}}'
```

对于 **<mirror_host_name>**，请指定您在镜像 registry 证书中指定的 registry 域名；对于 **<credentials>**，请指定您的镜像 registry 的 base64 编码用户名和密码。

b. 添加 **additionalTrustBundle** 参数和值。

```
additionalTrustBundle: |
 -----BEGIN CERTIFICATE-----

ZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZ
 -----END CERTIFICATE-----
```

该值必须是您用于镜像 registry 的证书文件内容，可以是现有的可信证书颁发机构或您为镜像 registry 生成的自签名证书。

c. 定义在父 **platform.gcp** 字段中安装集群的 VPC 的网络和子网：

```
network: <existing_vpc>
controlPlaneSubnet: <control_plane_subnet>
computeSubnet: <compute_subnet>
```

对于 **platform.gcp.network**，指定现有 Google VPC 的名称。对于 **platform.gcp.controlPlaneSubnet** 和 **platform.gcp.computeSubnet**，请分别指定要分别部署 control plane 机器和计算机器的现有子网。

d. 添加镜像内容资源，如下例所示：

```
imageContentSources:
- mirrors:
  - <mirror_host_name>:5000/<repo_name>/release
  source: quay.example.com/openshift-release-dev/ocp-release
- mirrors:
  - <mirror_host_name>:5000/<repo_name>/release
  source: registry.example.com/ocp/release
```

要完成这些值，请使用您在创建镜像容器镜像仓库（registry）时记录的 **imageContentSources**。

3. 对需要的 **install-config.yaml** 文件做任何其他修改。您可以在**安装配置参数**部分中找到有关可用参数的更多信息。

4. 备份 **install-config.yaml** 文件，以便用于安装多个集群。

重要

install-config.yaml 文件会在安装过程中消耗掉。如果要重复使用此文件，必须现在备份。

4.11.5.3. 在安装过程中配置集群范围代理

生产环境可能会拒绝直接访问互联网，而是提供 HTTP 或 HTTPS 代理。您可以通过在 **install-config.yaml** 文件中配置代理设置，将新的 OpenShift Container Platform 集群配置为使用代理。

先决条件

- 您有一个现有的 **install-config.yaml** 文件。
- 您检查了集群需要访问的站点，并决定是否需要绕过代理。默认情况下代理所有集群出口流量，包括对托管云供应商 API 的调用。您需要将站点添加到 **Proxy** 对象的 **spec.noProxy** 字段来绕过代理。

注意

Proxy 对象 **status.noProxy** 字段使用安装配置中的 **networking.machineNetwork[].cidr**、**networking.clusterNetwork[].cidr** 和 **networking.serviceNetwork[]** 字段的值填充。

对于在 Amazon Web Services(AWS)、Google Cloud Platform(GCP)、Microsoft Azure 和 Red Hat OpenStack Platform(RHOSP)上安装, **Proxy** 对象 **status.noProxy** 字段也会使用实例元数据端点填充(**169.254.169.254**)。

流程

1. 编辑 **install-config.yaml** 文件并添加代理设置。例如：

```
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: example.com 3
additionalTrustBundle: | 4
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
...
```

1. 用于创建集群外 HTTP 连接的代理 URL。URL 必须是 **http**。
2. 用于创建集群外 HTTPS 连接的代理 URL。
3. 要排除在代理中的目标域名、IP 地址或其他网络 CIDR 的逗号分隔列表。在域前面加 **.** 来仅匹配子域。例如： **.y.com** 匹配 **x.y.com**，但不匹配 **y.com**。使用 * 绕过所有目的地的代理。
4. 如果提供，安装程序会在 **openshift-config 命名空间中生成名为 user-ca- bundle** 的配置映射来保存额外的 CA 证书。如果您提供 **additionalTrustBundle** 和至少一个代理设置，则 **Proxy** 对象会被配置为引用 **trustedCA** 字段中的 **user-ca-bundle** 配置映射。然后，Cluster Network Operator 会创建一个 **trusted-ca-bundle** 配置映射，该配置映射将为 **trustedCA** 参数指定的内容与 RHCOS 信任捆绑包合并。 **additionalTrustBundle** 字段是必需的，除非代理的身份证书由来自 RHCOS 信任捆绑包的颁发机构签名。

注意

安装程序不支持代理的 **readinessEndpoints** 字段。

2. 保存该文件，并在安装 OpenShift Container Platform 时引用。

安装程序会创建一个名为 **cluster** 的集群范围代理，该代理使用提供的 **install-config.yaml** 文件中的代理设置。如果没有提供代理设置，仍然会创建一个 **cluster Proxy** 对象，但它会有一个空 **spec**。

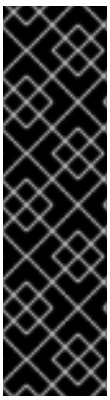
注意

只支持名为 **cluster** 的 **Proxy** 对象，且无法创建额外的代理。

4.11.5.4. 创建 Kubernetes 清单和 Ignition 配置文件

由于您必须修改一些集群定义文件并要手动启动集群机器，因此您必须生成 Kubernetes 清单和 Ignition 配置文件，集群需要这两项来创建其机器。

安装配置文件转换为 Kubernetes 清单。清单嵌套到 Ignition 配置文件中，稍后用于创建集群。



重要

- 安装程序生成的 Ignition 配置文件包含在 24 小时后过期的证书，然后在过期时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外情况是，您需要手动批准待处理的 **node-bootstrapper** 证书签名请求（CSR）来恢复 kubelet 证书。如需更多信息，请参阅*从过期的 control plane 证书中恢复*的文档。
- 建议您在生成 12 小时后使用 Ignition 配置文件，因为集群安装后 24 小时证书从 16 小时轮转至 22 小时。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中运行证书更新时避免安装失败。

先决条件

- 已获得 OpenShift Container Platform 安装程序。对于受限网络安装，这些文件位于您的堡垒主机上。
- 已创建 **install-config.yaml** 安装配置文件。

流程

1. 切换到包含安装程序的目录，并为集群生成 Kubernetes 清单：

```
$ ./openshift-install create manifests --dir <installation_directory> 1
```

1 对于 **<installation_directory>**，请指定含有您创建的 **install-config.yaml** 文件的安装目录。

2. 删除定义 control plane 机器的 Kubernetes 清单文件：

```
$ rm -f <installation_directory>/openshift/99_openshift-cluster-api_master-machines-*.yaml
```

通过删除这些文件，您可以防止集群自动生成 control plane 机器。

3. 可选：如果您不希望集群置备计算机器，请删除定义 worker 机器的 Kubernetes 清单文件：

```
$ rm -f <installation_directory>/openshift/99_openshift-cluster-api_worker-machineset-*.yaml
```

由于您要自行创建并管理 worker 机器，因此不需要初始化这些机器。

4. 检查 **<installation_directory>/manifests/cluster-scheduler-02-config.yml** Kubernetes 清单文件中的 **mastersSchedulable** 参数是否已设置为 **false**。此设置可防止在 control plane 机器上调度 pod:

   a. 打开 **<installation_directory>/manifests/cluster-scheduler-02-config.yml** 文件。

   b. 找到 **mastersSchedulable** 参数并确保它被设置为 **false**。

c. 保存并退出文件。

5. 可选：如果您不希望 Ingress Operator 代表您创建 DNS 记录，请删除 **<installation_directory>/manifests/cluster-dns-02-config.yml** DNS 配置文件中的 **privateZone** 和 **publicZone** 部分：

```
apiVersion: config.openshift.io/v1
kind: DNS
metadata:
  creationTimestamp: null
  name: cluster
spec:
  baseDomain: example.openshift.com
  privateZone: 1
    id: mycluster-100419-private-zone
  publicZone: 2
    id: example.openshift.com
status: {}
```

1 2 完全删除此部分。

如果您这样做，后续步骤中必须手动添加入口 DNS 记录。

6. 要创建 Ignition 配置文件，从包含安装程序的目录运行以下命令：

```
$ ./openshift-install create ignition-configs --dir <installation_directory> 1
```

1 对于 **<installation_directory>**，请指定相同的安装目录。

该目录中将生成以下文件：

```
.
├── auth
│   ├── kubeadmin-password
│   └── kubeconfig
├── bootstrap.ign
├── master.ign
├── metadata.json
└── worker.ign
```

其他资源

- 可选：添加入口 DNS 记录

4.11.6. 导出常用变量

4.11.6.1. 提取基础架构名称

Ignition 配置文件包含一个唯一集群标识符，您可以使用它在 Google Cloud Platform (GCP) 中唯一地标识您的集群。基础架构名称还用于在 OpenShift Container Platform 安装过程中定位适当的 GCP 资源。提供的 Deployment Manager 模板包含对此基础架构名称的引用，因此您必须提取它。

先决条件

- 获取 OpenShift Container Platform 安装程序以及集群的 pull secret。
- 已为集群生成 Ignition 配置文件。
- 安装了 **jq** 软件包。

流程

- 要从 Ignition 配置文件元数据中提取和查看基础架构名称，请运行以下命令：

```
$ jq -r .infraID <installation_directory>/metadata.json 1
```

1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

输出示例

```
openshift-vw9j6 1
```

1 此命令的输出是您的集群名称和随机字符串。

4.11.6.2. 为 Deployment Manager 模板导出常用变量

您必须导出与提供的 Deployment Manager 模板搭配使用的一组常用变量，它们有助于在 Google Cloud Platform (GCP) 上完成用户提供基础架构安装。

注意

特定的 Deployment Manager 模板可能还需要其他导出变量，这些变量在相关的程序中详细介绍。

先决条件

- 获取 OpenShift Container Platform 安装程序以及集群的 pull secret。
- 为集群生成 Ignition 配置文件。
- 安装 **jq** 软件包。

流程

1. 导出由提供的 Deployment Manager 模板使用的以下常用变量：

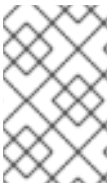
```
$ export BASE_DOMAIN='<base_domain>'
$ export BASE_DOMAIN_ZONE_NAME='<base_domain_zone_name>'
$ export NETWORK_CIDR='10.0.0.0/16'
$ export MASTER_SUBNET_CIDR='10.0.0.0/19'
$ export WORKER_SUBNET_CIDR='10.0.32.0/19'

$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
$ export CLUSTER_NAME=`jq -r .clusterName <installation_directory>/metadata.json`
```

```
$ export INFRA_ID=`jq -r .infraID <installation_directory>/metadata.json`
$ export PROJECT_NAME=`jq -r .gcp.projectID <installation_directory>/metadata.json`
$ export REGION=`jq -r .gcp.region <installation_directory>/metadata.json`
```

1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

4.11.7. 在 GCP 中创建 VPC

您必须在 Google Cloud Platform (GCP) 中创建一个 VPC，供您的 OpenShift Container Platform 集群使用。您可以自定义 VPC 来满足您的要求。创建 VPC 的一种方法是修改提供的 Deployment Manager 模板。



注意

如果不使用提供的 Deployment Manager 模板来创建 GCP 基础架构，您必须检查提供的信息并手动创建基础架构。如果集群没有正确初始化，您可能需要联系红帽支持并提供您的安装日志。

先决条件

- 配置 GCP 帐户。
- 为集群生成 Ignition 配置文件。

流程

1. 复制 **VPC 的 Deployment Manager 模板**一节中的模板，并将它以 **01_vpc.py** 形式保存到计算机上。此模板描述了集群所需的 VPC。
2. 创建 **01_vpc.yaml** 资源定义文件：

```
$ cat <<EOF >01_vpc.yaml
imports:
- path: 01_vpc.py

resources:
- name: cluster-vpc
  type: 01_vpc.py
  properties:
    infra_id: '${INFRA_ID}' 1
    region: '${REGION}' 2
    master_subnet_cidr: '${MASTER_SUBNET_CIDR}' 3
    worker_subnet_cidr: '${WORKER_SUBNET_CIDR}' 4
EOF
```

1 **infra_id** 是来自于提取步骤的 **INFRA_ID** 基础架构名称。

2 **region** 是集群要部署到的区域，如 **us-central1**。

3 **master_subnet_cidr** 是 master 子网的 CIDR，如 **10.0.0.0/19**。

4 **worker_subnet_cidr** 是 worker 子网的 CIDR，如 **10.0.32.0/19**。

3. 使用 **gcloud** CLI 创建部署：

```
$ gcloud deployment-manager deployments create ${INFRA_ID}-vpc --config 01_vpc.yaml
```

4.11.7.1. VPC 的 Deployment Manager 模板

您可以使用以下 Deployment Manager 模板来部署 OpenShift Container Platform 集群所需的 VPC：

例 4.19. 01_VPC.py Deployment Manager 模板

```
def GenerateConfig(context):

    resources = [{
        'name': context.properties['infra_id'] + '-network',
        'type': 'compute.v1.network',
        'properties': {
            'region': context.properties['region'],
            'autoCreateSubnetworks': False
        }
    }, {
        'name': context.properties['infra_id'] + '-master-subnet',
        'type': 'compute.v1.subnetwork',
        'properties': {
            'region': context.properties['region'],
            'network': '$(ref.' + context.properties['infra_id'] + '-network.selfLink)',
            'ipCidrRange': context.properties['master_subnet_cidr']
        }
    }, {
        'name': context.properties['infra_id'] + '-worker-subnet',
        'type': 'compute.v1.subnetwork',
        'properties': {
            'region': context.properties['region'],
            'network': '$(ref.' + context.properties['infra_id'] + '-network.selfLink)',
            'ipCidrRange': context.properties['worker_subnet_cidr']
        }
    }, {
        'name': context.properties['infra_id'] + '-router',
        'type': 'compute.v1.router',
        'properties': {
            'region': context.properties['region'],
            'network': '$(ref.' + context.properties['infra_id'] + '-network.selfLink)',
            'nats': [{
                'name': context.properties['infra_id'] + '-nat-master',
                'natIpAllocateOption': 'AUTO_ONLY',
                'minPortsPerVm': 7168,
                'sourceSubnetworkIpRangesToNat': 'LIST_OF_SUBNETWORKS',
                'subnetworks': [{
                    'name': '$(ref.' + context.properties['infra_id'] + '-master-subnet.selfLink)',
                    'sourceIpRangesToNat': ['ALL_IP_RANGES']
                }]
            }, {
                'name': context.properties['infra_id'] + '-nat-worker',
                'natIpAllocateOption': 'AUTO_ONLY',
                'minPortsPerVm': 512,
                'sourceSubnetworkIpRangesToNat': 'LIST_OF_SUBNETWORKS',
```

```
                'subnetworks': [{
                    'name': '$(ref.' + context.properties['infra_id'] + '-worker-subnet.selfLink)',
                    'sourceIpRangesToNat': ['ALL_IP_RANGES']
                }]
            }]
        }
    }]

    return {'resources': resources}
```

4.11.8. 用户置备的基础架构对网络的要求

所有 Red Hat Enterprise Linux CoreOS（RHCOS）机器在启动过程中需要 **initramfs** 中的网络从机器配置服务器获取 Ignition 配置。

您必须配置机器间的网络连接，以便集群组件进行通信。每台机器都必须能够解析集群中所有其他机器的主机名。

表 4.48. 所有机器到所有机器

协议	端口	描述
ICMP	N/A	网络可访问性测试
TCP	**1936**	指标
	9000-9999	主机级别的服务，包括端口 **9100-9101** 上的节点导出器和端口 **9099** 上的 Cluster Version Operator。
	10250-10259	Kubernetes 保留的默认端口
	10256	openshift-sdn
UDP	**4789**	VXLAN 和 Geneve
	6081	VXLAN 和 Geneve
	9000-9999	主机级别的服务，包括端口 **9100-9101** 上的节点导出器。
TCP/UDP	**30000-32767**	Kubernetes 节点端口

表 4.49. 要通过控制平面的所有机器

协议	端口	描述
TCP	**6443**	Kubernetes API

表 4.50. control plane 机器到 control plane 机器

协议	端口	描述
TCP	**2379-2380**	etcd 服务器和对等端口

网络拓扑要求
您为集群置备的基础架构必须满足下列网络拓扑要求。

负载均衡器
在安装 OpenShift Container Platform 前，您必须置备两个满足以下要求的负载均衡器：

1. **API 负载均衡器**：提供一个通用端点，供用户（包括人和机器）与平台交互和配置。配置以下条件：

   - 只适用于第 4 层负载均衡。这可被称为 Raw TCP、SSL Passthrough 或者 SSL 桥接模式。如果使用 SSL Bridge 模式，必须为 API 路由启用 Server Name Indication（SNI）。
   - 无状态负载平衡算法。这些选项根据负载均衡器的实现而有所不同。

   重要

   不要为 API 负载均衡器配置会话持久性。

   在负载均衡器的前端和后台配置以下端口：

   表 4.51. API 负载均衡器

端口	后端机器（池成员）	内部	外部	描述
6443	Bootstrap 和 control plane.bootstrap 机器初始化集群 control plane 后，您要从负载均衡器中删除 bootstrap 机器。您必须为 API 服务器健康检查探测配置 **/readyz** 端点。	X	X	Kubernetes API 服务器
22623	Bootstrap 和 control plane.bootstrap 机器初始化集群 control plane 后，您要从负载均衡器中删除 bootstrap 机器。	X		机器配置服务器

   注意

   负载均衡器必须配置为，从 API 服务器关闭 **/readyz** 端点到从池中删除 API 服务器实例时最多需要 30 秒。在 **/readyz** 返回错误或处于健康状态后的时间范围内，端点必须被删除或添加。每 5 秒或 10 秒探测一次，有两个成功请求处于健康状态，三个成为不健康的请求经过测试。

2. **应用程序入口负载均衡器**:提供来自集群外部的应用程序流量流量的 Ingress 点。配置以下条件：

   - 只适用于第 4 层负载均衡。这可被称为 Raw TCP、SSL Passthrough 或者 SSL 桥接模式。如果使用 SSL Bridge 模式，您必须为 Ingress 路由启用Server Name Indication（SNI）。

- 建议根据可用选项以及平台上托管的应用程序类型，使用基于连接的或者基于会话的持久性。

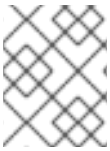
在负载均衡器的前端和后台配置以下端口：

表 4.52. 应用程序入口负载均衡器

端口	后端机器（池成员）	内部	外部	描述
443	默认运行入口路由器 Pod、计算或 worker 的机器。	X	X	HTTPS 流量
80	默认运行入口路由器 Pod、计算或 worker 的机器。	X	X	HTTP 流量

提示

如果负载均衡器可以看到客户端的真实 IP 地址，启用基于 IP 的会话持久性可提高使用端到端 TLS 加密的应用程序的性能。

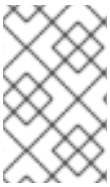


注意

OpenShift Container Platform 集群需要正确配置入口路由器。control plane 初始化后，您必须配置入口路由器。

4.11.9. 在 GCP 中创建负载均衡器

您必须在 Google Cloud Platform (GCP) 中配置负载均衡器，供您的 OpenShift Container Platform 集群使用。创建这些组件的一种方法是修改提供的 Deployment Manager 模板。



注意

如果不使用提供的 Deployment Manager 模板来创建 GCP 基础架构，您必须检查提供的信息并手动创建基础架构。如果集群没有正确初始化，您可能需要联系红帽支持并提供您的安装日志。

先决条件

- 配置 GCP 帐户。
- 为集群生成 Ignition 配置文件。
- 在 GCP 中创建和配置 VPC 及相关子网。

流程

1. 复制负载**均衡器的 Deployment Manager 模板**一节中的模板，并将它以 **02_lb_int.py** 形式保存到计算机上。此模板描述了集群所需的负载均衡对象。
2. 对于外部集群，还要将本主题的 **外部负载均衡器的 Deployment Manager 模板** 部分中的模板进行复制，并将它以 **02_lb_ext.py** 形式保存到计算机上。此模板描述了集群所需的外部负载均衡对象。

3. 导出部署模板使用的变量：

   a. 导出集群网络位置：

   ```
   $ export CLUSTER_NETWORK=(`gcloud compute networks describe ${INFRA_ID}-network --format json | jq -r .selfLink`)
   ```

   b. 导出 control plane 子网位置：

   ```
   $ export CONTROL_SUBNET=(`gcloud compute networks subnets describe ${INFRA_ID}-master-subnet --region=${REGION} --format json | jq -r .selfLink`)
   ```

   c. 导出集群使用的三个区域（zone）：

   ```
   $ export ZONE_0=(`gcloud compute regions describe ${REGION} --format=json | jq -r .zones[0] | cut -d "/" -f9`)
   ```

   ```
   $ export ZONE_1=(`gcloud compute regions describe ${REGION} --format=json | jq -r .zones[1] | cut -d "/" -f9`)
   ```

   ```
   $ export ZONE_2=(`gcloud compute regions describe ${REGION} --format=json | jq -r .zones[2] | cut -d "/" -f9`)
   ```

4. 创建 **02_infra.yaml** 资源定义文件：

```
$ cat <<EOF >02_infra.yaml
imports:
- path: 02_lb_ext.py
- path: 02_lb_int.py (1)
resources:
- name: cluster-lb-ext (2)
  type: 02_lb_ext.py
  properties:
    infra_id: '${INFRA_ID}' (3)
    region: '${REGION}' (4)
- name: cluster-lb-int
  type: 02_lb_int.py
  properties:
    cluster_network: '${CLUSTER_NETWORK}'
    control_subnet: '${CONTROL_SUBNET}' (5)
    infra_id: '${INFRA_ID}'
    region: '${REGION}'
    zones: (6)
    - '${ZONE_0}'
    - '${ZONE_1}'
    - '${ZONE_2}'
EOF
```

(1) (2) 仅在部署外部集群时需要。

(3) **infra_id** 是来自于提取步骤的 **INFRA_ID** 基础架构名称。

(4) **region** 是集群要部署到的区域，如 **us-central1**。

5 **control_subnet** 是到控制子网的 URI。

6 **zones** 是 control plane 实例要部署到的区域，如 **us-east1-b**、**us-east1-c** 和 **us-east1-d**。

5. 使用 **gcloud** CLI 创建部署：

```
$ gcloud deployment-manager deployments create ${INFRA_ID}-infra --config 02_infra.yaml
```

6. 导出集群 IP 地址：

```
$ export CLUSTER_IP=(`gcloud compute addresses describe ${INFRA_ID}-cluster-ip --
region=${REGION} --format json | jq -r .address`)
```

7. 对于外部集群，还要导出集群公共 IP 地址：

```
$ export CLUSTER_PUBLIC_IP=(`gcloud compute addresses describe ${INFRA_ID}-cluster-
public-ip --region=${REGION} --format json | jq -r .address`)
```

4.11.9.1. 外部负载均衡器的 Deployment Manager 模板

您可以使用以下 Deployment Manager 模板来部署 OpenShift Container Platform 集群所需的外部负载均衡器：

例 4.20. 02_lb_ext.py Deployment Manager 模板

```
def GenerateConfig(context):

    resources = [{
        'name': context.properties['infra_id'] + '-cluster-public-ip',
        'type': 'compute.v1.address',
        'properties': {
            'region': context.properties['region']
        }
    }, {
        # Refer to docs/dev/kube-apiserver-health-check.md on how to correctly setup health check
probe for kube-apiserver
        'name': context.properties['infra_id'] + '-api-http-health-check',
        'type': 'compute.v1.httpHealthCheck',
        'properties': {
            'port': 6080,
            'requestPath': '/readyz'
        }
    }, {
        'name': context.properties['infra_id'] + '-api-target-pool',
        'type': 'compute.v1.targetPool',
        'properties': {
            'region': context.properties['region'],
            'healthChecks': ['$(ref.' + context.properties['infra_id'] + '-api-http-health-check.selfLink)'],
            'instances': []
        }
    }, {
        'name': context.properties['infra_id'] + '-api-forwarding-rule',
        'type': 'compute.v1.forwardingRule',
```

```
        'properties': {
            'region': context.properties['region'],
            'IPAddress': '$(ref.' + context.properties['infra_id'] + '-cluster-public-ip.selfLink)',
            'target': '$(ref.' + context.properties['infra_id'] + '-api-target-pool.selfLink)',
            'portRange': '6443'
        }
    }]

    return {'resources': resources}
```

4.11.9.2. 内部负载均衡器的 Deployment Manager 模板

您可以使用以下 Deployment Manager 模板来部署 OpenShift Container Platform 集群所需的内部负载均衡器：

例 4.21. 02_lb_int.py Deployment Manager 模板

```
def GenerateConfig(context):

    backends = []
    for zone in context.properties['zones']:
        backends.append({
            'group': '$(ref.' + context.properties['infra_id'] + '-master-' + zone + '-instance-group' +
'.selfLink)'
        })

    resources = [{
        'name': context.properties['infra_id'] + '-cluster-ip',
        'type': 'compute.v1.address',
        'properties': {
            'addressType': 'INTERNAL',
            'region': context.properties['region'],
            'subnetwork': context.properties['control_subnet']
        }
    }, {
        # Refer to docs/dev/kube-apiserver-health-check.md on how to correctly setup health check
probe for kube-apiserver
        'name': context.properties['infra_id'] + '-api-internal-health-check',
        'type': 'compute.v1.healthCheck',
        'properties': {
            'httpsHealthCheck': {
                'port': 6443,
                'requestPath': '/readyz'
            },
            'type': "HTTPS"
        }
    }, {
        'name': context.properties['infra_id'] + '-api-internal-backend-service',
        'type': 'compute.v1.regionBackendService',
        'properties': {
            'backends': backends,
            'healthChecks': ['$(ref.' + context.properties['infra_id'] + '-api-internal-health-
check.selfLink)'],
            'loadBalancingScheme': 'INTERNAL',
```

```
            'region': context.properties['region'],
            'protocol': 'TCP',
            'timeoutSec': 120
        }
    }, {
        'name': context.properties['infra_id'] + '-api-internal-forwarding-rule',
        'type': 'compute.v1.forwardingRule',
        'properties': {
            'backendService': '$(ref.' + context.properties['infra_id'] + '-api-internal-backend-
service.selfLink)',
            'IPAddress': '$(ref.' + context.properties['infra_id'] + '-cluster-ip.selfLink)',
            'loadBalancingScheme': 'INTERNAL',
            'ports': ['6443','22623'],
            'region': context.properties['region'],
            'subnetwork': context.properties['control_subnet']
        }
    }]

    for zone in context.properties['zones']:
        resources.append({
            'name': context.properties['infra_id'] + '-master-' + zone + '-instance-group',
            'type': 'compute.v1.instanceGroup',
            'properties': {
                'namedPorts': [
                    {
                        'name': 'ignition',
                        'port': 22623
                    }, {
                        'name': 'https',
                        'port': 6443
                    }
                ],
                'network': context.properties['cluster_network'],
                'zone': zone
            }
        })

    return {'resources': resources}
```

创建外部集群时，除了 **02_lb_ext.py** 模板外，还需要此模板。

4.11.10. 在 GCP 中创建私有 DNS 区域

您必须在 Google Cloud Platform (GCP) 中配置一个私人的 DNS 区以供您的 OpenShift Container Platform 集群使用。创建此组件的一种方法是修改提供的 Deployment Manager 模板。

注意

如果不使用提供的 Deployment Manager 模板来创建 GCP 基础架构，您必须检查提供的信息并手动创建基础架构。如果集群没有正确初始化，您可能需要联系红帽支持并提供您的安装日志。

先决条件

- 配置 GCP 帐户。
- 为集群生成 Ignition 配置文件。
- 在 GCP 中创建和配置 VPC 及相关子网。

流程

1. 复制 **私人 DNS 的 Deployment Manager 模板**一节中的模板，并将它以 **02_dns.py** 形式保存到计算机上。此模板描述了集群所需的私有 DNS 对象。
2. 创建 **02_dns.yaml** 资源定义文件：

```
$ cat <<EOF >02_dns.yaml
imports:
- path: 02_dns.py

resources:
- name: cluster-dns
  type: 02_dns.py
  properties:
    infra_id: '${INFRA_ID}' 1
    cluster_domain: '${CLUSTER_NAME}.${BASE_DOMAIN}' 2
    cluster_network: '${CLUSTER_NETWORK}' 3
EOF
```

1. **infra_id** 是来自于提取步骤的 **INFRA_ID** 基础架构名称。
2. **cluster_domain** 是集群的域，如 **openshift.example.com**。
3. **cluster_network** 是集群网络的 **selfLink** URL。

3. 使用 **gcloud** CLI 创建部署：

```
$ gcloud deployment-manager deployments create ${INFRA_ID}-dns --config 02_dns.yaml
```

4. 由于 Deployment Manager 的限制，模板不会创建 DNS 条目，因此您必须手动创建它们：
   a. 添加内部 DNS 条目：

```
$ if [ -f transaction.yaml ]; then rm transaction.yaml; fi
$ gcloud dns record-sets transaction start --zone ${INFRA_ID}-private-zone
$ gcloud dns record-sets transaction add ${CLUSTER_IP} --name
api.${CLUSTER_NAME}.${BASE_DOMAIN}. --ttl 60 --type A --zone ${INFRA_ID}-
private-zone
$ gcloud dns record-sets transaction add ${CLUSTER_IP} --name api-
int.${CLUSTER_NAME}.${BASE_DOMAIN}. --ttl 60 --type A --zone ${INFRA_ID}-
private-zone
$ gcloud dns record-sets transaction execute --zone ${INFRA_ID}-private-zone
```

   b. 对于外部集群，还要添加外部 DNS 条目：

```
$ if [ -f transaction.yaml ]; then rm transaction.yaml; fi
$ gcloud dns record-sets transaction start --zone ${BASE_DOMAIN_ZONE_NAME}
```

```
$ gcloud dns record-sets transaction add ${CLUSTER_PUBLIC_IP} --name
api.${CLUSTER_NAME}.${BASE_DOMAIN}. --ttl 60 --type A --zone
${BASE_DOMAIN_ZONE_NAME}
$ gcloud dns record-sets transaction execute --zone ${BASE_DOMAIN_ZONE_NAME}
```

4.11.10.1. 私有 DNS 的 Deployment Manager 模板

您可以使用以下 Deployment Manager 模板来部署 OpenShift Container Platform 集群所需的 私有 DNS：

例 4.22. 02_dns.py Deployment Manager 模板

```
def GenerateConfig(context):

    resources = [{
        'name': context.properties['infra_id'] + '-private-zone',
        'type': 'dns.v1.managedZone',
        'properties': {
            'description': '',
            'dnsName': context.properties['cluster_domain'] + '.',
            'visibility': 'private',
            'privateVisibilityConfig': {
                'networks': [{
                    'networkUrl': context.properties['cluster_network']
                }]
            }
        }
    }]

    return {'resources': resources}
```

4.11.11. 在 GCP 中创建防火墙规则

您必须在 Google Cloud Platform (GCP) 中创建一个防火墙，供您的 OpenShift Container Platform 集群使用。创建这些组件的一种方法是修改提供的 Deployment Manager 模板。

注意

如果不使用提供的 Deployment Manager 模板来创建 GCP 基础架构，您必须检查提供的信息并手动创建基础架构。如果集群没有正确初始化，您可能需要联系红帽支持并提供您的安装日志。

先决条件

- 配置 GCP 帐户。
- 为集群生成 Ignition 配置文件。
- 在 GCP 中创建和配置 VPC 及相关子网。

流程

1. 复制**防火墙规则的 Deployment Manager 模板**一节中的模板，并将它以 **03_firewall.py** 形式保存到计算机上。此模板描述了集群所需的安全组。

2. 创建 **03_firewall.yaml** 资源定义文件：

```
$ cat <<EOF >03_firewall.yaml
imports:
- path: 03_firewall.py

resources:
- name: cluster-firewall
  type: 03_firewall.py
  properties:
    allowed_external_cidr: '0.0.0.0/0' (1)
    infra_id: '${INFRA_ID}' (2)
    cluster_network: '${CLUSTER_NETWORK}' (3)
    network_cidr: '${NETWORK_CIDR}' (4)
EOF
```

(1) **allowed_external_cidr** 是 CIDR 范围，可访问集群 API 和 SSH 到 bootstrap 主机。对于内部集群，将此值设置为 **${NETWORK_CIDR}**。

(2) **infra_id** 是来自于提取步骤的 **INFRA_ID** 基础架构名称。

(3) **cluster_network** 是集群网络的 **selfLink** URL。

(4) **network_cidr** 是 VPC 网络的 CIDR，如 **10.0.0.0/16**。

3. 使用 **gcloud** CLI 创建部署：

```
$ gcloud deployment-manager deployments create ${INFRA_ID}-firewall --config
03_firewall.yaml
```

4.11.11.1. 防火墙规则的 Deployment Manager 模板

您可以使用以下 Deployment Manager 模板来部署 OpenShift Container Platform 集群所需的防火墙规则：

例 4.23. 03_firewall.py Deployment Manager 模板

```
def GenerateConfig(context):

    resources = [{
        'name': context.properties['infra_id'] + '-bootstrap-in-ssh',
        'type': 'compute.v1.firewall',
        'properties': {
            'network': context.properties['cluster_network'],
            'allowed': [{
                'IPProtocol': 'tcp',
                'ports': ['22']
            }],
            'sourceRanges': [context.properties['allowed_external_cidr']],
            'targetTags': [context.properties['infra_id'] + '-bootstrap']
        }
```

```
    }, {
        'name': context.properties['infra_id'] + '-api',
        'type': 'compute.v1.firewall',
        'properties': {
            'network': context.properties['cluster_network'],
            'allowed': [{
                'IPProtocol': 'tcp',
                'ports': ['6443']
            }],
            'sourceRanges': [context.properties['allowed_external_cidr']],
            'targetTags': [context.properties['infra_id'] + '-master']
        }
    }, {
        'name': context.properties['infra_id'] + '-health-checks',
        'type': 'compute.v1.firewall',
        'properties': {
            'network': context.properties['cluster_network'],
            'allowed': [{
                'IPProtocol': 'tcp',
                'ports': ['6080', '6443', '22624']
            }],
            'sourceRanges': ['35.191.0.0/16', '130.211.0.0/22', '209.85.152.0/22', '209.85.204.0/22'],
            'targetTags': [context.properties['infra_id'] + '-master']
        }
    }, {
        'name': context.properties['infra_id'] + '-etcd',
        'type': 'compute.v1.firewall',
        'properties': {
            'network': context.properties['cluster_network'],
            'allowed': [{
                'IPProtocol': 'tcp',
                'ports': ['2379-2380']
            }],
            'sourceTags': [context.properties['infra_id'] + '-master'],
            'targetTags': [context.properties['infra_id'] + '-master']
        }
    }, {
        'name': context.properties['infra_id'] + '-control-plane',
        'type': 'compute.v1.firewall',
        'properties': {
            'network': context.properties['cluster_network'],
            'allowed': [{
                'IPProtocol': 'tcp',
                'ports': ['10257']
            },{
                'IPProtocol': 'tcp',
                'ports': ['10259']
            },{
                'IPProtocol': 'tcp',
                'ports': ['22623']
            }],
            'sourceTags': [
                context.properties['infra_id'] + '-master',
                context.properties['infra_id'] + '-worker'
            ],
            'targetTags': [context.properties['infra_id'] + '-master']
```

```
        }
    }, {
        'name': context.properties['infra_id'] + '-internal-network',
        'type': 'compute.v1.firewall',
        'properties': {
            'network': context.properties['cluster_network'],
            'allowed': [{
                'IPProtocol': 'icmp'
            },{
                'IPProtocol': 'tcp',
                'ports': ['22']
            }],
            'sourceRanges': [context.properties['network_cidr']],
            'targetTags': [
                context.properties['infra_id'] + '-master',
                context.properties['infra_id'] + '-worker'
            ]
        }
    }, {
        'name': context.properties['infra_id'] + '-internal-cluster',
        'type': 'compute.v1.firewall',
        'properties': {
            'network': context.properties['cluster_network'],
            'allowed': [{
                'IPProtocol': 'udp',
                'ports': ['4789', '6081']
            },{
                'IPProtocol': 'tcp',
                'ports': ['9000-9999']
            },{
                'IPProtocol': 'udp',
                'ports': ['9000-9999']
            },{
                'IPProtocol': 'tcp',
                'ports': ['10250']
            },{
                'IPProtocol': 'tcp',
                'ports': ['30000-32767']
            },{
                'IPProtocol': 'udp',
                'ports': ['30000-32767']
            }],
            'sourceTags': [
                context.properties['infra_id'] + '-master',
                context.properties['infra_id'] + '-worker'
            ],
            'targetTags': [
                context.properties['infra_id'] + '-master',
                context.properties['infra_id'] + '-worker'
            ]
        }
    }]

    return {'resources': resources}
```

4.11.12. 在 GCP 中创建 IAM 角色

您必须在 Google Cloud Platform (GCP) 中创建一个 IAM 角色，供您的 OpenShift Container Platform 集群使用。创建这些组件的一种方法是修改提供的 Deployment Manager 模板。

注意

如果不使用提供的 Deployment Manager 模板来创建 GCP 基础架构，您必须检查提供的信息并手动创建基础架构。如果集群没有正确初始化，您可能需要联系红帽支持并提供您的安装日志。

先决条件

- 配置 GCP 帐户。
- 为集群生成 Ignition 配置文件。
- 在 GCP 中创建和配置 VPC 及相关子网。

流程

1. 复制 **IAM 角色的 Deployment Manager 模板**一节中的模板，并将它以 **03_iam.p** 形式保存到计算机上。此模板描述了集群所需的 IAM 角色。
2. 创建 **03_iam.yaml** 资源定义文件：

```
$ cat <<EOF >03_iam.yaml
imports:
- path: 03_iam.py
resources:
- name: cluster-iam
  type: 03_iam.py
  properties:
    infra_id: '${INFRA_ID}' (1)
EOF
```

(1) **infra_id** 是来自于提取步骤的 **INFRA_ID** 基础架构名称。

3. 使用 **gcloud** CLI 创建部署：

```
$ gcloud deployment-manager deployments create ${INFRA_ID}-iam --config 03_iam.yaml
```

4. 导出 master 服务帐户的变量：

```
$ export MASTER_SERVICE_ACCOUNT=(`gcloud iam service-accounts list --filter
"email~^${INFRA_ID}-m@${PROJECT_NAME}." --format json | jq -r '.[0].email'`)
```

5. 导出 worker 服务帐户的变量：

```
$ export WORKER_SERVICE_ACCOUNT=(`gcloud iam service-accounts list --filter
"email~^${INFRA_ID}-w@${PROJECT_NAME}." --format json | jq -r '.[0].email'`)
```

6. 导出托管计算机器的子网的变量：

```
$ export COMPUTE_SUBNET=(`gcloud compute networks subnets describe ${INFRA_ID}-
worker-subnet --region=${REGION} --format json | jq -r .selfLink`)
```

7. 由于 Deployment Manager 的限制，模板不会创建策略绑定，因此您必须手动创建它们：

```
$ gcloud projects add-iam-policy-binding ${PROJECT_NAME} --member
"serviceAccount:${MASTER_SERVICE_ACCOUNT}" --role "roles/compute.instanceAdmin"
$ gcloud projects add-iam-policy-binding ${PROJECT_NAME} --member
"serviceAccount:${MASTER_SERVICE_ACCOUNT}" --role "roles/compute.networkAdmin"
$ gcloud projects add-iam-policy-binding ${PROJECT_NAME} --member
"serviceAccount:${MASTER_SERVICE_ACCOUNT}" --role "roles/compute.securityAdmin"
$ gcloud projects add-iam-policy-binding ${PROJECT_NAME} --member
"serviceAccount:${MASTER_SERVICE_ACCOUNT}" --role "roles/iam.serviceAccountUser"
$ gcloud projects add-iam-policy-binding ${PROJECT_NAME} --member
"serviceAccount:${MASTER_SERVICE_ACCOUNT}" --role "roles/storage.admin"

$ gcloud projects add-iam-policy-binding ${PROJECT_NAME} --member
"serviceAccount:${WORKER_SERVICE_ACCOUNT}" --role "roles/compute.viewer"
$ gcloud projects add-iam-policy-binding ${PROJECT_NAME} --member
"serviceAccount:${WORKER_SERVICE_ACCOUNT}" --role "roles/storage.admin"
```

8. 创建服务帐户密钥，并将它保存到本地备用：

```
$ gcloud iam service-accounts keys create service-account-key.json --iam-
account=${MASTER_SERVICE_ACCOUNT}
```

4.11.12.1. IAM 角色的 Deployment Manager 模板

您可以使用以下 Deployment Manager 模板来部署 OpenShift Container Platform 集群所需的 IAM 角色：

例 4.24. 03_iam.py Deployment Manager 模板

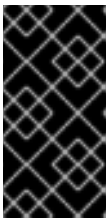
```
def GenerateConfig(context):

    resources = [{
        'name': context.properties['infra_id'] + '-master-node-sa',
        'type': 'iam.v1.serviceAccount',
        'properties': {
            'accountId': context.properties['infra_id'] + '-m',
            'displayName': context.properties['infra_id'] + '-master-node'
        }
    }, {
        'name': context.properties['infra_id'] + '-worker-node-sa',
        'type': 'iam.v1.serviceAccount',
        'properties': {
            'accountId': context.properties['infra_id'] + '-w',
            'displayName': context.properties['infra_id'] + '-worker-node'
        }
    }]

    return {'resources': resources}
```

4.11.13. 为 GCP 基础架构创建 RHCOS 集群镜像

您必须对 OpenShift Container Platform 节点的 Google Cloud Platform (GCP) 使用有效的 Red Hat Enterprise Linux CoreOS (RHCOS) 镜像。

流程

1. 从 RHCOS 镜像镜像页面获取 RHCOS 镜像。



重要

RHCOS 镜像可能不会随着 OpenShift Container Platform 的每一发行版本都有改变。您必须下载一个最高版本的镜像，其版本号应小于或等于您安装的 OpenShift Container Platform 版本。如果可用，请使用与 OpenShift Container Platform 版本匹配的镜像版本。

文件名包含 OpenShift Container Platform 版本号，格式为 **rhcos-<version>-<arch>-gcp.<arch>.tar.gz**。

2. 创建 Google 存储桶：

```
$ gsutil mb gs://<bucket_name>
```

3. 将 RHCOS 镜像上传到 Google 存储桶：

```
$ gsutil cp <downloaded_image_file_path>/rhcos-<version>-x86_64-gcp.x86_64.tar.gz
gs://<bucket_name>
```

4. 将上传的 RHCOS 镜像位置导出为变量：

```
$ export IMAGE_SOURCE="gs://<bucket_name>/rhcos-<version>-x86_64-
gcp.x86_64.tar.gz"
```

5. 创建集群镜像：

```
$ gcloud compute images create "${INFRA_ID}-rhcos-image" \
    --source-uri="${IMAGE_SOURCE}"
```

4.11.14. 在 GCP 中创建 bootstrap 机器

您必须在 Google Cloud Platform (GCP) 中创建 bootstrap 机器，以便在 OpenShift Container Platform 集群初始化过程中使用。创建此机器的一种方法是修改提供的 Deployment Manager 模板。

注意

如果不使用提供的 Deployment Manager 模板来创建 bootstrap 机器，您必须检查提供的信息并手动创建基础架构。如果集群没有正确初始化，您可能需要联系红帽支持并提供您的安装日志。

先决条件

- 配置 GCP 帐户。

- 为集群生成 Ignition 配置文件。
- 在 GCP 中创建和配置 VPC 及相关子网。
- 在 GCP 中创建和配置联网及负载均衡器。
- 创建 control plane 和计算角色。
- 确定安装了 pyOpenSSL。

流程

1. 复制 **bootstrap 机器的 Deployment Manager 模板**一节中的模板，并将它以 **04_bootstrap.py** 形式保存到计算机上。此模板描述了集群所需的 bootstrap 机器。
2. 导出安装程序所需的 Red Hat Enterprise Linux CoreOS（RHCOS）镜像的位置：

```
$ export CLUSTER_IMAGE=(`gcloud compute images describe ${INFRA_ID}-rhcos-image --format json | jq -r .selfLink`)
```

3. 创建存储桶并上传 **bootstrap.ign** 文件：

```
$ gsutil mb gs://${INFRA_ID}-bootstrap-ignition
$ gsutil cp <installation_directory>/bootstrap.ign gs://${INFRA_ID}-bootstrap-ignition/
```

4. 为 bootstrap 实例创建一个签名的 URL，用于访问 Ignition 配置。将输出中的 URL 导出为变量：

```
$ export BOOTSTRAP_IGN=`gsutil signurl -d 1h service-account-key.json gs://${INFRA_ID}-bootstrap-ignition/bootstrap.ign | grep "^gs:" | awk '{print $5}'`
```

5. 创建 **04_bootstrap.yaml** 资源定义文件：

```
$ cat <<EOF >04_bootstrap.yaml
imports:
- path: 04_bootstrap.py

resources:
- name: cluster-bootstrap
  type: 04_bootstrap.py
  properties:
    infra_id: '${INFRA_ID}' 1
    region: '${REGION}' 2
    zone: '${ZONE_0}' 3

    cluster_network: '${CLUSTER_NETWORK}' 4
    control_subnet: '${CONTROL_SUBNET}' 5
    image: '${CLUSTER_IMAGE}' 6
    machine_type: 'n1-standard-4' 7
    root_volume_size: '128' 8

    bootstrap_ign: '${BOOTSTRAP_IGN}' 9
EOF
```

1. **infra_id** 是来自于提取步骤的 **INFRA_ID** 基础架构名称。
2. **region** 是集群要部署到的区域，如 **us-central1**。
3. **zone** 是 Bootstrap 实例要部署到的区域，如 **us-central1-b**。
4. **cluster_network** 是集群网络的 **selfLink** URL。
5. **control_subnet** 是控制子网的 **selfLink** URL。
6. **image** 是 RHCOS 镜像的 **selfLink** URL。
7. **machine_type** 是实例的机器类型，如 **n1-standard-4**。
8. **root_volume_size** 是 bootstrap 机器的引导磁盘大小。
9. **bootstrap_ign** 是创建签名 URL 时的 URL 输出。

6. 使用 **gcloud** CLI 创建部署：

```
$ gcloud deployment-manager deployments create ${INFRA_ID}-bootstrap --config
04_bootstrap.yaml
```

7. 由于 Deployment Manager 的限制，模板无法管理负载均衡器成员资格，因此您必须手动添加 bootstrap 机器：

   a. 将 bootstrap 实例添加到内部负载均衡器实例组中：

```
$ gcloud compute instance-groups unmanaged add-instances \
    ${INFRA_ID}-bootstrap-instance-group --zone=${ZONE_0} --instances=${INFRA_ID}-
bootstrap
```

   b. 将 bootstrap 实例组添加到内部负载均衡器后端服务中：

```
$ gcloud compute backend-services add-backend \
    ${INFRA_ID}-api-internal-backend-service --region=${REGION} --instance-
group=${INFRA_ID}-bootstrap-instance-group --instance-group-zone=${ZONE_0}
```

4.11.14.1. bootstrap 机器的 Deployment Manager 模板

您可以使用以下 Deployment Manager 模板来部署 OpenShift Container Platform 集群所需的 bootstrap 机器：

例 4.25. 04_bootstrap.py Deployment Manager 模板

```
def GenerateConfig(context):

  resources = [{
    'name': context.properties['infra_id'] + '-bootstrap-public-ip',
    'type': 'compute.v1.address',
    'properties': {
        'region': context.properties['region']
    }
  }, {
```

```
        'name': context.properties['infra_id'] + '-bootstrap',
        'type': 'compute.v1.instance',
        'properties': {
            'disks': [{
                'autoDelete': True,
                'boot': True,
                'initializeParams': {
                    'diskSizeGb': context.properties['root_volume_size'],
                    'sourceImage': context.properties['image']
                }
            }],
            'machineType': 'zones/' + context.properties['zone'] + '/machineTypes/' +
context.properties['machine_type'],
            'metadata': {
                'items': [{
                    'key': 'user-data',
                    'value': '{"ignition":{"config":{"replace":{"source":"' + context.properties['bootstrap_ign']
+ '"}},"version":"3.1.0"}}',
                }]
            },
            'networkInterfaces': [{
                'subnetwork': context.properties['control_subnet'],
                'accessConfigs': [{
                    'natIP': '$(ref.' + context.properties['infra_id'] + '-bootstrap-public-ip.address)'
                }]
            }],
            'tags': {
                'items': [
                    context.properties['infra_id'] + '-master',
                    context.properties['infra_id'] + '-bootstrap'
                ]
            },
            'zone': context.properties['zone']
        }
    }, {
        'name': context.properties['infra_id'] + '-bootstrap-instance-group',
        'type': 'compute.v1.instanceGroup',
        'properties': {
            'namedPorts': [
                {
                    'name': 'ignition',
                    'port': 22623
                }, {
                    'name': 'https',
                    'port': 6443
                }
            ],
            'network': context.properties['cluster_network'],
            'zone': context.properties['zone']
        }
    }]

    return {'resources': resources}
```

4.11.15. 在 GCP 中创建 control plane 机器

您必须在 Google Cloud Platform (GCP) 中创建 control plane 机器，供您的集群使用。创建这些机器的一种方法是修改提供的 Deployment Manager 模板。

注意

如果不使用提供的 Deployment Manager 模板来创建 control plane 机器，您必须检查提供的信息并手动创建基础架构。如果集群没有正确初始化，您可能需要联系红帽支持并提供您的安装日志。

先决条件

- 配置 GCP 帐户。
- 为集群生成 Ignition 配置文件。
- 在 GCP 中创建和配置 VPC 及相关子网。
- 在 GCP 中创建和配置联网及负载均衡器。
- 创建 control plane 和计算角色。
- 创建 bootstrap 机器。

流程

1. 复制 **control plane 机器的 Deployment Manager 模板**一节中的模板，并将它以 **05_control_plane.py** 形式保存到计算机上。此模板描述了集群所需的 control plane 机器。
2. 导出资源定义所需的以下变量：

```
$ export MASTER_IGNITION=`cat <installation_directory>/master.ign`
```

3. 创建 **05_control_plane.yaml** 资源定义文件：

```
$ cat <<EOF >05_control_plane.yaml
imports:
- path: 05_control_plane.py

resources:
- name: cluster-control-plane
  type: 05_control_plane.py
  properties:
    infra_id: '${INFRA_ID}' (1)
    zones: (2)
    - '${ZONE_0}'
    - '${ZONE_1}'
    - '${ZONE_2}'

    control_subnet: '${CONTROL_SUBNET}' (3)
    image: '${CLUSTER_IMAGE}' (4)
    machine_type: 'n1-standard-4' (5)
    root_volume_size: '128'
```

```
    service_account_email: '${MASTER_SERVICE_ACCOUNT}' 6

    ignition: '${MASTER_IGNITION}' 7
EOF
```

1. **infra_id** 是来自于提取步骤的 **INFRA_ID** 基础架构名称。
2. **zones** 是 control plane 实例要部署到的区域，如 **us-central1-a**、**us-central1-b** 和 **us-central1-c**。
3. **control_subnet** 是控制子网的 **selfLink** URL。
4. **image** 是 RHCOS 镜像的 **selfLink** URL。
5. **machine_type** 是实例的机器类型，如 **n1-standard-4**。
6. **service_account_email** 是创建的 master 服务帐户的电子邮件地址。
7. **ignition** 是 **master.ign** 文件的内容。

4. 使用 **gcloud** CLI 创建部署：

```
$ gcloud deployment-manager deployments create ${INFRA_ID}-control-plane --config
05_control_plane.yaml
```

5. 由于 Deployment Manager 的限制，模板无法管理负载均衡器成员资格，因此您必须手动添加 control plane 机器。
   - 运行以下命令，将 control plane 机器添加到适当的实例组中：

```
$ gcloud compute instance-groups unmanaged add-instances ${INFRA_ID}-master-
${ZONE_0}-instance-group --zone=${ZONE_0} --instances=${INFRA_ID}-master-0
$ gcloud compute instance-groups unmanaged add-instances ${INFRA_ID}-master-
${ZONE_1}-instance-group --zone=${ZONE_1} --instances=${INFRA_ID}-master-1
$ gcloud compute instance-groups unmanaged add-instances ${INFRA_ID}-master-
${ZONE_2}-instance-group --zone=${ZONE_2} --instances=${INFRA_ID}-master-2
```

   - 对于外部集群，还需要运行以下命令将 control plane 机器添加到目标池中：

```
$ gcloud compute target-pools add-instances ${INFRA_ID}-api-target-pool --instances-
zone="${ZONE_0}" --instances=${INFRA_ID}-master-0
$ gcloud compute target-pools add-instances ${INFRA_ID}-api-target-pool --instances-
zone="${ZONE_1}" --instances=${INFRA_ID}-master-1
$ gcloud compute target-pools add-instances ${INFRA_ID}-api-target-pool --instances-
zone="${ZONE_2}" --instances=${INFRA_ID}-master-2
```

4.11.15.1. control plane 机器的 Deployment Manager 模板

您可以使用以下 Deployment Manager 模板来部署 OpenShift Container Platform 集群所需的control plane 机器：

例 4.26. 05_control_plane.py Deployment Manager 模板

```
def GenerateConfig(context):

    resources = [{
        'name': context.properties['infra_id'] + '-master-0',
        'type': 'compute.v1.instance',
        'properties': {
            'disks': [{
                'autoDelete': True,
                'boot': True,
                'initializeParams': {
                    'diskSizeGb': context.properties['root_volume_size'],
                    'diskType': 'zones/' + context.properties['zones'][0] + '/diskTypes/pd-ssd',
                    'sourceImage': context.properties['image']
                }
            }],
            'machineType': 'zones/' + context.properties['zones'][0] + '/machineTypes/' +
context.properties['machine_type'],
            'metadata': {
                'items': [{
                    'key': 'user-data',
                    'value': context.properties['ignition']
                }]
            },
            'networkInterfaces': [{
                'subnetwork': context.properties['control_subnet']
            }],
            'serviceAccounts': [{
                'email': context.properties['service_account_email'],
                'scopes': ['https://www.googleapis.com/auth/cloud-platform']
            }],
            'tags': {
                'items': [
                    context.properties['infra_id'] + '-master',
                ]
            },
            'zone': context.properties['zones'][0]
        }
    }, {
        'name': context.properties['infra_id'] + '-master-1',
        'type': 'compute.v1.instance',
        'properties': {
            'disks': [{
                'autoDelete': True,
                'boot': True,
                'initializeParams': {
                    'diskSizeGb': context.properties['root_volume_size'],
                    'diskType': 'zones/' + context.properties['zones'][1] + '/diskTypes/pd-ssd',
                    'sourceImage': context.properties['image']
                }
            }],
            'machineType': 'zones/' + context.properties['zones'][1] + '/machineTypes/' +
context.properties['machine_type'],
            'metadata': {
                'items': [{
                    'key': 'user-data',
                    'value': context.properties['ignition']
```

```
            }]
        },
        'networkInterfaces': [{
            'subnetwork': context.properties['control_subnet']
        }],
        'serviceAccounts': [{
            'email': context.properties['service_account_email'],
            'scopes': ['https://www.googleapis.com/auth/cloud-platform']
        }],
        'tags': {
            'items': [
                context.properties['infra_id'] + '-master',
            ]
        },
        'zone': context.properties['zones'][1]
    }
}, {
    'name': context.properties['infra_id'] + '-master-2',
    'type': 'compute.v1.instance',
    'properties': {
        'disks': [{
            'autoDelete': True,
            'boot': True,
            'initializeParams': {
                'diskSizeGb': context.properties['root_volume_size'],
                'diskType': 'zones/' + context.properties['zones'][2] + '/diskTypes/pd-ssd',
                'sourceImage': context.properties['image']
            }
        }],
        'machineType': 'zones/' + context.properties['zones'][2] + '/machineTypes/' +
context.properties['machine_type'],
        'metadata': {
            'items': [{
                'key': 'user-data',
                'value': context.properties['ignition']
            }]
        },
        'networkInterfaces': [{
            'subnetwork': context.properties['control_subnet']
        }],
        'serviceAccounts': [{
            'email': context.properties['service_account_email'],
            'scopes': ['https://www.googleapis.com/auth/cloud-platform']
        }],
        'tags': {
            'items': [
                context.properties['infra_id'] + '-master',
            ]
        },
        'zone': context.properties['zones'][2]
    }
}]

return {'resources': resources}
```

4.11.16. 等待 bootstrap 完成并删除 GCP 中的 bootstrap 资源

在 Google Cloud Platform (GCP) 中创建所有所需的基础架构后，请等待您通过安装程序生成的 Ignition 配置文件所置备的机器上完成 bootstrap 过程。

先决条件

- 配置 GCP 帐户。
- 为集群生成 Ignition 配置文件。
- 在 GCP 中创建和配置 VPC 及相关子网。
- 在 GCP 中创建和配置联网及负载均衡器。
- 创建 control plane 和计算角色。
- 创建 bootstrap 机器。
- 创建 control plane 机器。

流程

1. 更改到包含安装程序的目录，再运行以下命令：

```
$ ./openshift-install wait-for bootstrap-complete --dir <installation_directory> \ 1
    --log-level info 2
```

1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

2 要查看不同的安装详情，请指定 **warn**、**debug** 或 **error**，而不要指定 **info**。

如果命令退出且不显示 **FATAL** 警告，则代表您的 control plane 已被初始化。

2. 删除 bootstrap 资源：

```
$ gcloud compute backend-services remove-backend ${INFRA_ID}-api-internal-backend-
service --region=${REGION} --instance-group=${INFRA_ID}-bootstrap-instance-group --
instance-group-zone=${ZONE_0}
$ gsutil rm gs://${INFRA_ID}-bootstrap-ignition/bootstrap.ign
$ gsutil rb gs://${INFRA_ID}-bootstrap-ignition
$ gcloud deployment-manager deployments delete ${INFRA_ID}-bootstrap
```

4.11.17. 在 GCP 中创建额外的 worker 机器

您可以通过分散启动各个实例或利用集群外自动化流程（如自动缩放组），在 Google Cloud Platform (GCP) 中为您的集群创建 worker 机器。您还可以利用 OpenShift Container Platform 中的内置集群扩展机制和机器 API。

在本例中，您要使用 Deployment Manager 模板来手动启动一个实例。通过在文件中添加类型为 **06_worker.py** 的其他资源，即可启动其他实例。

注意

如果不使用提供的 Deployment Manager 模板来创建 worker 机器，您必须检查提供的信息并手动创建基础架构。如果集群没有正确初始化，您可能需要联系红帽支持并提供您的安装日志。

先决条件

- 配置 GCP 帐户。
- 为集群生成 Ignition 配置文件。
- 在 GCP 中创建和配置 VPC 及相关子网。
- 在 GCP 中创建和配置联网及负载均衡器。
- 创建 control plane 和计算角色。
- 创建 bootstrap 机器。
- 创建 control plane 机器。

流程

1. 复制本主题的 **worker 机器的 Deployment Manager 模板**部分中的模板，并将它以 **06_worker.py** 形式保存到计算机中。此模板描述了集群所需的 worker 机器。
2. 导出资源定义使用的变量。
   a. 导出托管计算机器的子网：

   ```
   $ export COMPUTE_SUBNET=(`gcloud compute networks subnets describe
   ${INFRA_ID}-worker-subnet --region=${REGION} --format json | jq -r .selfLink`)
   ```

   b. 为您的服务帐户导出电子邮件地址：

   ```
   $ export WORKER_SERVICE_ACCOUNT=(`gcloud iam service-accounts list --filter
   "email~^${INFRA_ID}-w@${PROJECT_NAME}." --format json | jq -r '.[0].email'`)
   ```

   c. 导出计算机器 Ignition 配置文件的位置：

   ```
   $ export WORKER_IGNITION=`cat <installation_directory>/worker.ign`
   ```

3. 创建 **06_worker.yaml** 资源定义文件：

```
$ cat <<EOF >06_worker.yaml
imports:
- path: 06_worker.py

resources:
- name: 'worker-0' (1)
  type: 06_worker.py
  properties:
    infra_id: '${INFRA_ID}' (2)
    zone: '${ZONE_0}' (3)
```

```
    compute_subnet: '${COMPUTE_SUBNET}' 4
    image: '${CLUSTER_IMAGE}' 5
    machine_type: 'n1-standard-4' 6
    root_volume_size: '128'
    service_account_email: '${WORKER_SERVICE_ACCOUNT}' 7
    ignition: '${WORKER_IGNITION}' 8
- name: 'worker-1'
  type: 06_worker.py
  properties:
    infra_id: '${INFRA_ID}' 9
    zone: '${ZONE_1}' 10
    compute_subnet: '${COMPUTE_SUBNET}' 11
    image: '${CLUSTER_IMAGE}' 12
    machine_type: 'n1-standard-4' 13
    root_volume_size: '128'
    service_account_email: '${WORKER_SERVICE_ACCOUNT}' 14
    ignition: '${WORKER_IGNITION}' 15
EOF
```

1 **name** 是 worker 机器的名称，如 **worker-0**。

2 9 **infra_id** 是来自于提取步骤的 **INFRA_ID** 基础架构名称。

3 10 **zone** 是 worker 机器要部署到的区域，如 **us-central1-a**。

4 11 **compute_subnet** 是计算子网的 **selfLink**。

5 12 **image** 是 RHCOS 镜像的 **selfLink** URL。

6 13 **machine_type** 是实例的机器类型，如 **n1-standard-4**。

7 14 **service_account_email** 是创建的 worker 服务帐户的电子邮件地址。

8 15 **ignition** 是 **worker.ign** 文件的内容。

4. 可选：如果要启动更多实例，请在 **06_worker.yaml** 资源定义文件中包含类型为 **06_worker.py** 的其他资源。

5. 使用 **gcloud** CLI 创建部署：

```
$ gcloud deployment-manager deployments create ${INFRA_ID}-worker --config
06_worker.yaml
```

4.11.17.1. worker 机器的 Deloyment Manager 模板

您可以使用以下 Deployment Manager 模板来部署 OpenShift Container Platform 集群所需的 worker 机器：

例 4.27. 06_worker.py Deployment Manager 模板

```
def GenerateConfig(context):

    resources = [{
```

```
        'name': context.properties['infra_id'] + '-' + context.env['name'],
        'type': 'compute.v1.instance',
        'properties': {
            'disks': [{
                'autoDelete': True,
                'boot': True,
                'initializeParams': {
                    'diskSizeGb': context.properties['root_volume_size'],
                    'sourceImage': context.properties['image']
                }
            }],
            'machineType': 'zones/' + context.properties['zone'] + '/machineTypes/' +
context.properties['machine_type'],
            'metadata': {
                'items': [{
                    'key': 'user-data',
                    'value': context.properties['ignition']
                }]
            },
            'networkInterfaces': [{
                'subnetwork': context.properties['compute_subnet']
            }],
            'serviceAccounts': [{
                'email': context.properties['service_account_email'],
                'scopes': ['https://www.googleapis.com/auth/cloud-platform']
            }],
            'tags': {
                'items': [
                    context.properties['infra_id'] + '-worker',
                ]
            },
            'zone': context.properties['zone']
        }
    }]

    return {'resources': resources}
```

4.11.18. 使用 CLI 登录到集群

您可以通过导出集群 **kubeconfig** 文件，以默认系统用户身份登录集群。**kubeconfig** 文件包含关于集群的信息，供 CLI 用于将客户端连接到正确集群和 API 服务器。该文件特只适用于一个特定的集群，在 OpenShift Container Platform 安装过程中创建。

先决条件

- 已部署了 OpenShift Container Platform 集群。
- 已安装 **oc** CLI。

流程

1. 导出 **kubeadmin** 凭证：

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

2. 使用导出的配置，验证能否成功运行 **oc** 命令：

```
$ oc whoami
```

输出示例

```
system:admin
```

4.11.19. 禁用默认的 OperatorHub 源

在 OpenShift Container Platform 安装过程中，默认为 OperatorHub 配置由红帽和社区项目提供的源内容的 operator 目录。在受限网络环境中，必须以集群管理员身份禁用默认目录。

流程

- 通过在 **OperatorHub** 对象中添加 **disableAllDefaultSources: true** 来禁用默认目录的源：

```
$ oc patch OperatorHub cluster --type json \
    -p '[{"op": "add", "path": "/spec/disableAllDefaultSources", "value": true}]'
```

提示

或者，您可以使用 Web 控制台管理目录源。在 **Administration → Cluster Settings → Global Configuration → OperatorHub** 页面中，点 **Sources** 选项卡，其中可创建、删除、禁用和启用单独的源。

4.11.20. 批准机器的证书签名请求

将机器添加到集群时，会为您添加的每台机器生成两个待处理证书签名请求（CSR）。您必须确认这些 CSR 已获得批准，或根据需要自行批准。客户端请求必须首先被批准，然后是服务器请求。

先决条件

- 您已将机器添加到集群中。

流程

1. 确认集群可以识别这些机器：

```
$ oc get nodes
```

输出示例

```
NAME     STATUS   ROLES   AGE  VERSION
master-0 Ready    master  63m  v1.19.0
master-1 Ready    master  63m  v1.19.0
master-2 Ready    master  64m  v1.19.0
```

输出将列出您创建的所有机器。

注意

在一些 CSR 被批准前，以上输出可能不包括计算节点（也称为 worker 节点）。

2. 检查待处理的 CSR，并确保可以看到添加到集群中的每台机器都有 **Pending** 或 **Approved** 状态的客户端请求：

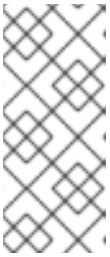
```
$ oc get csr
```

输出示例

```
NAME        AGE    REQUESTOR                                                   CONDITION
csr-8b2br   15m    system:serviceaccount:openshift-machine-config-operator:node-bootstrapper   Pending
csr-8vnps   15m    system:serviceaccount:openshift-machine-config-operator:node-bootstrapper   Pending
...
```

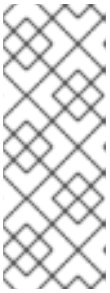
在本例中，两台机器加入了集群。您可能会在列表中看到更多已批准的 CSR。

3. 如果 CSR 没有获得批准，请在所添加机器的所有待处理 CSR 都处于 **Pending** 状态后，为您的集群机器批准这些 CSR：



注意

由于 CSR 会自动轮转，因此请在将机器添加到集群后一小时内批准您的 CSR。如果没有在一小时内批准，证书将会轮转，每个节点将会存在多个证书。您必须批准所有这些证书。批准客户端 CSR 后， Kubelet 为服务证书创建辅助 CSR，这需要手动批准。然后，如果 Kubelet 请求具有相同参数的新证书，则 **machine-approver** 会自动批准后续服务证书续订请求。



注意

对于在未启用机器 API 的平台中运行的集群，如裸机和其他用户置备的基础架构，必须采用一种方法自动批准 kubelet 提供证书请求（CSR）。如果没有批准请求，则 **oc exec**、**oc rsh** 和 **oc logs** 命令将无法成功，因为 API 服务器连接到 kubelet 时需要服务证书。与 Kubelet 端点联系的任何操作都需要此证书批准。这个方法必须监视新的 CSR，确认 CSR 由 **system:node** 或 **system:admin** 组中的 **node-bootstrapper** 服务帐户提交，并确认节点的身份。

- 若要单独批准，请对每个有效的 CSR 运行以下命令：

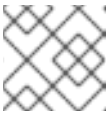
```
$ oc adm certificate approve <csr_name> 1
```

1 **<csr_name>** 是当前 CSR 列表中 CSR 的名称。

- 要批准所有待处理的 CSR，请运行以下命令：

```
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}{{"\n"}}{{end}}{{end}}' | xargs --no-run-if-empty oc adm certificate approve
```

-



注意

在有些 CSR 被批准前，一些 Operator 可能无法使用。

4. 现在，您的客户端请求已被批准，您必须查看添加到集群中的每台机器的服务器请求：

```
$ oc get csr
```

输出示例

```
NAME        AGE     REQUESTOR                                                   CONDITION
csr-bfd72   5m26s   system:node:ip-10-0-50-126.us-east-2.compute.internal       Pending
csr-c57lv   5m26s   system:node:ip-10-0-95-157.us-east-2.compute.internal       Pending
...
```

5. 如果剩余的 CSR 没有被批准，且处于 **Pending** 状态，请批准集群机器的 CSR：

- 若要单独批准，请对每个有效的 CSR 运行以下命令：

```
$ oc adm certificate approve <csr_name> 1
```

1 **<csr_name>** 是当前 CSR 列表中 CSR 的名称。

- 要批准所有待处理的 CSR，请运行以下命令：

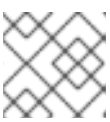
```
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}{{"\n"}}
{{end}}{{end}}' | xargs oc adm certificate approve
```

6. 批准所有客户端和服务器 CSR 后，器将处于 **Ready** 状态。运行以下命令验证：

```
$ oc get nodes
```

输出示例

```
NAME      STATUS    ROLES   AGE  VERSION
master-0  Ready     master  73m  v1.20.0
master-1  Ready     master  73m  v1.20.0
master-2  Ready     master  74m  v1.20.0
worker-0  Ready     worker  11m  v1.20.0
worker-1  Ready     worker  11m  v1.20.0
```



注意

批准服务器 CSR 后可能需要几分钟时间让机器转换为 **Ready** 状态。

其他信息

- 如需有关 CSR 的更多信息，请参阅证书签名请求。

4.11.21. 可选：添加入口 DNS 记录

如果您在创建 Kubernetes 清单并生成 Ignition 配置时删除了 DNS 区配置，您必须手动创建指向入口负载均衡器的 DNS 记录。您可以创建通配符 ***.apps.{baseDomain}.** 或具体的记录。您可以根据自己的要求使用 A、CNAME 和其他记录。

先决条件

- 配置 GCP 帐户。
- 在创建 Kubernetes 清单并生成 Ignition 配置时删除 DNS 区配置。
- 在 GCP 中创建和配置 VPC 及相关子网。
- 在 GCP 中创建和配置联网及负载均衡器。
- 创建 control plane 和计算角色。
- 创建 bootstrap 机器。
- 创建 control plane 机器。
- 创建 worker 机器。

流程

1. 等待入口路由器创建负载均衡器并填充 **EXTERNAL-IP** 字段：

```
$ oc -n openshift-ingress get service router-default
```

输出示例

```
NAME            TYPE           CLUSTER-IP     EXTERNAL-IP     PORT(S)                      AGE
router-default  LoadBalancer  172.30.18.154  35.233.157.184
80:32288/TCP,443:31215/TCP   98
```

2. 在您的区中添加 A 记录：

   - 使用 A 记录：

     i. 为路由器 IP 地址导出变量：

```
$ export ROUTER_IP=`oc -n openshift-ingress get service router-default --no-
headers | awk '{print $4}'`
```

     ii. 在专用区中添加 A 记录：

```
$ if [ -f transaction.yaml ]; then rm transaction.yaml; fi
$ gcloud dns record-sets transaction start --zone ${INFRA_ID}-private-zone
$ gcloud dns record-sets transaction add ${ROUTER_IP} --name
\*.apps.${CLUSTER_NAME}.${BASE_DOMAIN}. --ttl 300 --type A --zone
${INFRA_ID}-private-zone
$ gcloud dns record-sets transaction execute --zone ${INFRA_ID}-private-zone
```

     iii. 对于外部集群，还要将 A 记录添加到公共区：

```
$ if [ -f transaction.yaml ]; then rm transaction.yaml; fi
$ gcloud dns record-sets transaction start --zone ${BASE_DOMAIN_ZONE_NAME}
$ gcloud dns record-sets transaction add ${ROUTER_IP} --name
\*.apps.${CLUSTER_NAME}.${BASE_DOMAIN}. --ttl 300 --type A --zone
${BASE_DOMAIN_ZONE_NAME}
$ gcloud dns record-sets transaction execute --zone
${BASE_DOMAIN_ZONE_NAME}
```

- 如果需要添加特定域而不使用通配符，可以为集群的每个当前路由创建条目：

```
$ oc get --all-namespaces -o jsonpath='{range .items[*]}{range .status.ingress[*]}{.host}
{"\n"}{end}{end}' routes
```

输出示例

```
oauth-openshift.apps.your.cluster.domain.example.com
console-openshift-console.apps.your.cluster.domain.example.com
downloads-openshift-console.apps.your.cluster.domain.example.com
alertmanager-main-openshift-monitoring.apps.your.cluster.domain.example.com
grafana-openshift-monitoring.apps.your.cluster.domain.example.com
prometheus-k8s-openshift-monitoring.apps.your.cluster.domain.example.com
```

4.11.22. 在用户置备的基础架构上完成 GCP 安装

在 Google Cloud Platform (GCP) 用户置备的基础架构上启动 OpenShift Container Platform 安装后，您可以监控集群事件，直到集群就绪可用。

先决条件

- 在用户置备的 GCP 基础架构上为 OpenShift Container Platform 集群部署 bootstrap 机器。
- 安装 **oc** CLI 并登录。

流程

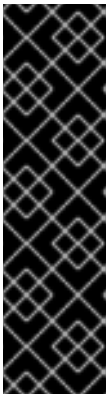
1. 完成集群安装：

```
$ ./openshift-install --dir <installation_directory> wait-for install-complete 1
```

输出示例

```
INFO Waiting up to 30m0s for the cluster to initialize...
```

1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。



重要

- 安装程序生成的 Ignition 配置文件包含在 24 小时后过期的证书，然后在过期时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外情况是，您需要手动批准待处理的 **node-bootstrapper** 证书签名请求（CSR）来恢复 kubelet 证书。如需更多信息，请参阅*从过期的 control plane 证书中恢复*的文档。

- 建议您在生成 12 小时后使用 Ignition 配置文件，因为集群安装后 24 小时证书从 16 小时轮转至 22 小时。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中运行证书更新时避免安装失败。

2. 观察集群的运行状态。

   a. 运行以下命令来查看当前的集群版本和状态：

   ```
   $ oc get clusterversion
   ```

   输出示例

   ```
   NAME     VERSION   AVAILABLE   PROGRESSING   SINCE   STATUS
   version            False       True          24m     Working towards 4.5.4: 99% complete
   ```

   b. 运行以下命令，查看 control plane 上由 Cluster Version Operator (CVO) 管理的 Operator：

   ```
   $ oc get clusteroperators
   ```

   输出示例

   ```
   NAME                                       VERSION   AVAILABLE   PROGRESSING   DEGRADED
   SINCE
   authentication                             4.5.4     True        False         False      7m56s
   cloud-credential                           4.5.4     True        False         False      31m
   cluster-autoscaler                         4.5.4     True        False         False      16m
   console                                    4.5.4     True        False         False      10m
   csi-snapshot-controller                    4.5.4     True        False         False      16m
   dns                                        4.5.4     True        False         False      22m
   etcd                                       4.5.4     False       False         False      25s
   image-registry                             4.5.4     True        False         False      16m
   ingress                                    4.5.4     True        False         False      16m
   insights                                   4.5.4     True        False         False      17m
   kube-apiserver                             4.5.4     True        False         False      19m
   kube-controller-manager                    4.5.4     True        False         False      20m
   kube-scheduler                             4.5.4     True        False         False      20m
   kube-storage-version-migrator              4.5.4     True        False         False      16m
   machine-api                                4.5.4     True        False         False      22m
   machine-config                             4.5.4     True        False         False      22m
   marketplace                                4.5.4     True        False         False      16m
   monitoring                                 4.5.4     True        False         False      10m
   network                                    4.5.4     True        False         False      23m
   node-tuning                                4.5.4     True        False         False      23m
   openshift-apiserver                        4.5.4     True        False         False      17m
   openshift-controller-manager               4.5.4     True        False         False      15m
   openshift-samples                          4.5.4     True        False         False      16m
   ```

```
operator-lifecycle-manager                 4.5.4     True        False         False      22m
operator-lifecycle-manager-catalog         4.5.4     True        False         False      22m
operator-lifecycle-manager-packageserver   4.5.4     True        False         False      18m
service-ca                                 4.5.4     True        False         False      23m
service-catalog-apiserver                  4.5.4     True        False         False      23m
service-catalog-controller-manager         4.5.4     True        False         False      23m
storage                                    4.5.4     True        False         False      17m
```

c. 运行以下命令来查看您的集群 pod：

```
$ oc get pods --all-namespaces
```

输出示例

```
NAMESPACE                                       NAME
READY     STATUS     RESTARTS   AGE
kube-system                                     etcd-member-ip-10-0-3-111.us-east-
2.compute.internal              1/1       Running     0          35m
kube-system                                     etcd-member-ip-10-0-3-239.us-east-
2.compute.internal              1/1       Running     0          37m
kube-system                                     etcd-member-ip-10-0-3-24.us-east-
2.compute.internal              1/1       Running     0          35m
openshift-apiserver-operator                    openshift-apiserver-operator-6d6674f4f4-
h7t2t                           1/1       Running     1          37m
openshift-apiserver                             apiserver-fm48r
1/1       Running     0          30m
openshift-apiserver                             apiserver-fxkvv
1/1       Running     0          29m
openshift-apiserver                             apiserver-q85nm
1/1       Running     0          29m
...
openshift-service-ca-operator                   openshift-service-ca-operator-66ff6dc6cd-
9r257                           1/1       Running     0          37m
openshift-service-ca                            apiservice-cabundle-injector-695b6bcbc-cl5hm
1/1       Running     0          35m
openshift-service-ca                            configmap-cabundle-injector-8498544d7-
25qn6                           1/1       Running     0          35m
openshift-service-ca                            service-serving-cert-signer-6445fc9c6-wqdqn
1/1       Running     0          35m
openshift-service-catalog-apiserver-operator    openshift-service-catalog-apiserver-
operator-549f44668b-b5q2w       1/1       Running     0          32m
openshift-service-catalog-controller-manager-operator   openshift-service-catalog-
controller-manager-operator-b78cr2lnm     1/1       Running     0          31m
```

当前集群版本是 **AVAILABLE** 时表示安装完毕。

4.11.23. OpenShift Container Platform 的 Telemetry 访问

在 OpenShift Container Platform 4.6 中，默认运行的 Telemetry 服务提供有关集群健康状况和成功更新的指标，需要访问互联网。如果您的集群连接到互联网，Telemetry 会自动运行，而且集群会注册到 OpenShift Cluster Manager。

确认 OpenShift Cluster Manager 清单正确后，可以由 Telemetry 自动维护，也可以使用 OpenShift Cluster Manager 手动维护，使用订阅监控来跟踪帐户或多集群级别的 OpenShift Container Platform 订阅。

其他资源

- 有关 Telemetry 服务的更多信息，请参阅关于远程健康监控。

4.11.24. 后续步骤

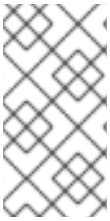
- 自定义集群。
- 为 Cluster Samples Operator 和 **must-gather** 工具配置镜像流。
- 了解如何在受限网络中使用 Operator Lifecycle Manager（OLM）。
- 如果您用来安装集群的镜像 registry 具有一个可信任的 CA，通过配置额外的信任存储将其添加到集群中。
- 如果需要，您可以选择不使用远程健康报告。

4.12. 在 GCP 上卸载集群

您可以删除部署到 Google Cloud Platform (GCP) 的集群。

4.12.1. 删除使用安装程序置备的基础架构的集群

您可以从云中删除使用安装程序置备的基础架构的集群。



注意

卸载后，检查云供应商是否有没有被正确移除的资源，特别是 User Provisioned Infrastructure（UPI）集群。可能存在安装程序没有创建的资源，或者安装程序无法访问的资源。例如，有些 Google Cloud 资源需要在共享 VPC 主机项目中有 IAM 权限， 或者可能存在必须删除的未使用健康检查。

先决条件

- 有部署集群时所用的安装程序副本。
- 有创建集群时安装程序所生成的文件。

流程

1. 在用来安装集群的计算机中包含安装程序的目录中，运行以下命令：

```
$ ./openshift-install destroy cluster \
--dir <installation_directory> --log-level info 1 2
```

1. 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。
2. 要查看不同的详情，请指定 **warn**、**debug** 或 **error**，而不要指定 **info**。

注意

您必须为集群指定包含集群定义文件的目录。安装程序需要此目录中的 **metadata.json** 文件来删除集群。

2. 可选：删除 **<installation_directory>** 目录和 OpenShift Container Platform 安装程序。

第 5 章 在裸机上安装

5.1. 在裸机上安装集群

在 OpenShift Container Platform 版本 4.6 中，您可以在您置备的裸机基础架构上安装集群。

重要

虽然您可能能够按照此流程在虚拟化或云环境中部署集群，但您必须清楚非裸机平台的其他注意事项。在尝试在此类环境中安装 OpenShift Container Platform 集群前，请参阅有关在未经测试的平台上部署 OpenShift Container Platform 的指南中的信息。

5.1.1. 先决条件

- 查看有关 OpenShift Container Platform 安装和更新流程的详细信息。
- 如果使用防火墙，则必须将其配置为允许集群需要访问的站点。

注意

如果您要配置代理，请务必也要查看此站点列表。

5.1.2. OpenShift Container Platform 的互联网访问

在 OpenShift Container Platform 4.6 中，您需要访问互联网来安装集群。

您必须具有以下互联网访问权限：

- 访问 OpenShift Cluster Manager 以下载安装程序并执行订阅管理。如果集群可以访问互联网，并且没有禁用 Telemetry，该服务会自动授权您的集群。
- 访问 Quay.io，以获取安装集群所需的软件包。
- 获取执行集群更新所需的软件包。

重要

如果您的集群无法直接访问互联网，则可以在置备的某些类基础架构上执行受限网络安装。在此过程中，您要下载所需的内容，并使用它在镜像 registry（mirror registry） 中填充安装集群并生成安装程序所需的软件包。对于某些安装类型，集群要安装到的环境不需要访问互联网。在更新集群之前，要更新 registry 镜像系统中的内容。

5.1.3. 具有用户置备基础架构的集群的机器要求

对于含有用户置备的基础架构的集群，您必须部署所有所需的机器。

5.1.3.1. 所需的机器

最小的 OpenShift Container Platform 集群需要下列主机：

- 一个临时 bootstrap 机器
- 三台 control plane 或 master 机器

- 至少两台计算机器，也称为 worker 机器。如果您正在运行三节点集群，则支持运行零个计算机器。不支持运行一台计算机器。

注意

集群要求 bootstrap 机器在三台 control plane 机器上部署 OpenShift Container Platform 集群。您可在安装集群后删除 bootstrap 机器。

重要

要保持集群的高可用性，请将独立的物理主机用于这些集群机器。

bootstrap 和 control plane 机器必须使用 Red Hat Enterprise Linux CoreOS (RHCOS) 作为操作系统。但是，计算机器可以在 Red Hat Enterprise Linux CoreOS(RHCOS)或 Red Hat Enterprise Linux(RHEL)7.9 之间进行选择。

请注意，RHCOS 基于 Red Hat Enterprise Linux（RHEL） 8，并继承其所有硬件认证和要求。请查看Red Hat Enterprise Linux 技术功能及限制 。

5.1.3.2. 网络连接要求

所有 Red Hat Enterprise Linux CoreOS (RHCOS) 机器在启动过程中需要 **initramfs** 中的网络从 Machine Config Server 获取 Ignition 配置文件。在初次启动过程中，需要一个 DHCP 服务器或设置了静态 IP 地址来建立网络连接，以下载它们的 Ignition 配置文件。另外，集群中的每个 OpenShift Container Platform 节点都必须有权访问网络时间协议（NTP）服务器。如果 DHCP 服务器提供 NTP 服务器信息，Red Hat Enterprise Linux CoreOS（RHCOS）机器上的 chrony 时间服务会读取信息，并可与 NTP 服务器同步时钟。

5.1.3.3. 最低资源要求

每台集群机器都必须满足以下最低要求：

表 5.1. 最低资源要求

机器	操作系统	CPU [1]	RAM	存储	IOPS [2]
bootstrap	RHCOS	4	16 GB	100 GB	300
Control plane	RHCOS	4	16 GB	100 GB	300
Compute	RHCOS 或 RHEL 7.9	2	8 GB	100 GB	300

1. 当未启用并发多线程(SMT)或超线程时，一个 CPU 相当于一个物理内核。启用后，使用以下公式来计算对应的比率：（每个内核的线程数）[1] 插槽 = CPU。

2. OpenShift Container Platform 和 Kubernetes 对磁盘性能非常敏感，建议使用更快的存储速度，特别是 control plane 节点上需要 10 ms p99 fsync 持续时间的 etcd。请注意，在许多云平台上，存储大小和 IOPS 可一起扩展，因此您可能需要过度分配存储卷来获取足够的性能。

5.1.3.4. 证书签名请求管理

在使用您置备的基础架构时，集群只能有限地访问自动机器管理，因此您必须提供一种在安装后批准集群证书签名请求 (CSR) 的机制。**kube-controller-manager** 只能批准 kubelet 客户端 CSR。 **machine-approver** 无法保证使用 kubelet 凭证请求的提供证书的有效性，因为它不能确认是正确的机器发出了该请求。您必须决定并实施一种方法，以验证 kubelet 提供证书请求的有效性并进行批准。

5.1.4. 创建用户置备的基础架构

在部署采用用户置备的基础架构的 OpenShift Container Platform 集群前，您必须创建底层基础架构。

先决条件

- 在为集群创建支持基础架构之前，请参阅OpenShift Container Platform 4.x Tested Integrations页。

流程

1. 在每个节点上配置 DHCP 或设置静态 IP 地址。
2. 提供所需的负载均衡器。
3. 配置机器的端口。
4. 配置 DNS。
5. 确保网络可以正常工作。

5.1.4.1. 用户置备的基础架构对网络的要求

所有 Red Hat Enterprise Linux CoreOS（RHCOS）机器在启动过程中需要 **initramfs** 中的网络从机器配置服务器获取 Ignition 配置。

在初次启动过程中，需要一个 DHCP 服务器或集群中的每个机器都设置了静态 IP 地址来建立网络连接，以下载它们的 Ignition 配置文件。

建议您使用 DHCP 服务器为集群进行长期机器管理。确保 DHCP 服务器已配置为向集群机器提供持久 IP 地址和主机名。

Kubernetes API 服务器必须能够解析集群机器的节点名称。如果 API 服务器和 worker 节点位于不同的区域中，您可以配置默认 DNS 搜索区域，以便 API 服务器能够解析节点名称。另一种支持的方法是始终在节点对象和所有 DNS 请求中使用完全限定域名来指代主机。

您必须配置机器间的网络连接，以便集群组件进行通信。每台机器都必须能够解析集群中所有其他机器的主机名。

表 5.2. 所有机器到所有机器

协议	端口	描述
ICMP	N/A	网络可访问性测试
TCP	**1936**	指标
	9000-**9999**	主机级别的服务，包括端口 **9100**-**9101** 上的节点导出器和端口 **9099** 上的 Cluster Version Operator。

协议	端口	描述
	10250-10259	Kubernetes 保留的默认端口
	10256	openshift-sdn
UDP	**4789**	VXLAN 和 Geneve
	6081	VXLAN 和 Geneve
	9000-9999	主机级别的服务，包括端口 **9100-9101** 上的节点导出器。
TCP/UDP	**30000-32767**	Kubernetes 节点端口

表 5.3. 要通过控制平面的所有机器

协议	端口	描述
TCP	**6443**	Kubernetes API

表 5.4. control plane 机器到 control plane 机器

协议	端口	描述
TCP	**2379-2380**	etcd 服务器和对等端口

网络拓扑要求

您为集群置备的基础架构必须满足下列网络拓扑要求。

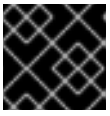
重要

OpenShift Container Platform 要求所有节点都能访问互联网，以便为平台容器提取镜像并向红帽提供遥测数据。

负载均衡器

在安装 OpenShift Container Platform 前，您必须置备两个满足以下要求的负载均衡器：

1. **API 负载均衡器**：提供一个通用端点，供用户（包括人和机器）与平台交互和配置。配置以下条件：
   - 只适用于第 4 层负载均衡。这可被称为 Raw TCP、SSL Passthrough 或者 SSL 桥接模式。如果使用 SSL Bridge 模式，必须为 API 路由启用 Server Name Indication（SNI）。
   - 无状态负载平衡算法。这些选项根据负载均衡器的实现而有所不同。



重要

不要为 API 负载均衡器配置会话持久性。

在负载均衡器的前端和后台配置以下端口：

表 5.5. API 负载均衡器

端口	后端机器（池成员）	内部	外部	描述
6443	Bootstrap 和 control plane.bootstrap 机器初始化集群 control plane 后，您要从负载均衡器中删除 bootstrap 机器。您必须为 API 服务器健康检查探测配置 **/readyz** 端点。	X	X	Kubernetes API 服务器
22623	Bootstrap 和 control plane.bootstrap 机器初始化集群 control plane 后，您要从负载均衡器中删除 bootstrap 机器。	X		机器配置服务器

注意

负载均衡器必须配置为，从 API 服务器关闭 **/readyz** 端点到从池中删除 API 服务器实例时最多需要 30 秒。在 **/readyz** 返回错误或处于健康状态后的时间范围内，端点必须被删除或添加。每 5 秒或 10 秒探测一次，有两个成功请求处于健康状态，三个成为不健康的请求经过测试。

2. **应用程序入口负载均衡器**:提供来自集群外部的应用程序流量流量的 Ingress 点。配置以下条件：

   - 只适用于第 4 层负载均衡。这可被称为 Raw TCP、SSL Passthrough 或者 SSL 桥接模式。如果使用 SSL Bridge 模式，您必须为 Ingress 路由启用Server Name Indication（SNI）。
   - 建议根据可用选项以及平台上托管的应用程序类型，使用基于连接的或者基于会话的持久性。

   在负载均衡器的前端和后台配置以下端口：

   表 5.6. 应用程序入口负载均衡器

端口	后端机器（池成员）	内部	外部	描述
443	默认运行入口路由器 Pod、计算或 worker 的机器。	X	X	HTTPS 流量
80	默认运行入口路由器 Pod、计算或 worker 的机器。	X	X	HTTP 流量

提示

如果负载均衡器可以看到客户端的真实 IP 地址，启用基于 IP 的会话持久性可提高使用端到端 TLS 加密的应用程序的性能。

注意

OpenShift Container Platform 集群需要正确配置入口路由器。control plane 初始化后，您必须配置入口路由器。

NTP 配置

OpenShift Container Platform 集群默认配置为使用公共网络时间协议（NTP）服务器。如果要使用本地企业 NTP 服务器，或者集群部署在断开连接的网络中，您可以将集群配置为使用特定的时间服务器。如需更多信息，请参阅*配置 chrony 时间服务*的文档。

如果 DHCP 服务器提供 NTP 服务器信息，Red Hat Enterprise Linux CoreOS（RHCOS）机器上的 chrony 时间服务会读取信息，并可与 NTP 服务器同步时钟。

其他资源

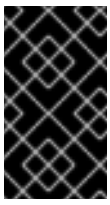
- 配置 chrony 时间服务

5.1.4.2. 用户置备 DNS 要求

DNS 用于名称解析和反向名称解析。DNS A/AAAA 或 CNAME 记录用于名称解析，PTR 记录用于反向解析名称。反向记录很重要，因为 Red Hat Enterprise Linux CoreOS（RHCOS）使用反向记录为所有节点设置主机名。另外，反向记录用于生成 OpenShift Container Platform 需要操作的证书签名请求（CSR）。

采用用户置备的基础架构的 OpenShift Container Platform 集群需要以下 DNS 记录。在每一记录中，**<cluster_name>** 是集群名称，**<base_domain>** 则是您在 **install-config.yaml** 文件中指定的集群基域。完整的 DNS 记录采用如下格式: **<component>.<cluster_name>.<base_domain>.**。

表 5.7. 所需的 DNS 记录

组件	记录	描述
Kubernetes API	**api.<cluster_name>.<base_domain>.**	添加 DNS A/AAAA 或 CNAME 记录，以及 DNS PTR 记录，以识别 control plane 机器的负载均衡器。这些记录必须由集群外的客户端以及集群中的所有节点解析。
	api-int.<cluster_name>.<base_domain>.	添加 DNS A/AAAA 或 CNAME 记录，以及 DNS PTR 记录，以识别 control plane 机器的负载均衡器。这些记录必须可以从集群中的所有节点解析。  **重要** API 服务器必须能够根据在 Kubernetes 中记录的主机名解析 worker 节点。如果 API 服务器无法解析节点名称，则代理的 API 调用会失败，且您无法从 pod 检索日志。
Routes	***.apps.<cluster_name>.<base_domain>.**	添加通配符 DNS A/AAAA 或 CNAME 记录，指向以运行入口路由器 Pod 的机器（默认为 worker 节点）为目标的负载均衡器。这些记录必须由集群外的客户端以及集群中的所有节点解析。

组件	记录	描述
bootstrap	**bootstrap.<cluster_name>.<base_domain>.**	添加 DNS A/AAAA 或 CNAME 记录，以及 DNS PTR 记录来识别 bootstrap 机器。这些记录必须由集群中的节点解析。
Master 主机	**<master><n>.<cluster_name>.<base_domain>.**	DNS A/AAAA 或 CNAME 记录，以识别 control plane 节点（也称为 master 节点）的每台机器。这些记录必须由集群中的节点解析。
Worker 主机	**<worker><n>.<cluster_name>.<base_domain>.**	添加 DNS A/AAAA 或 CNAME 记录，以识别 worker 节点的每台机器。这些记录必须由集群中的节点解析。

提示

您可以使用 **nslookup <hostname>** 命令来验证名称解析。您可以使用 **dig -x <ip_address>** 命令来验证 PTR 记录的反向名称解析。

下面的 BIND 区文件的例子展示了关于名字解析的 A 记录的例子。这个示例的目的是显示所需的记录。这个示例不是为选择一个名称解析服务提供建议。

例 5.1. DNS 区数据库示例

```
$TTL 1W
@ IN SOA ns1.example.com. root (
   2019070700 ; serial
   3H  ; refresh (3 hours)
   30M  ; retry (30 minutes)
   2W  ; expiry (2 weeks)
   1W )  ; minimum (1 week)
 IN NS ns1.example.com.
 IN MX 10 smtp.example.com.
;
;
ns1 IN A 192.168.1.5
smtp IN A 192.168.1.5
;
helper IN A 192.168.1.5
helper.ocp4 IN A 192.168.1.5
;
; The api identifies the IP of your load balancer.
api.ocp4  IN A 192.168.1.5
api-int.ocp4  IN A 192.168.1.5
;
; The wildcard also identifies the load balancer.
*.apps.ocp4  IN A 192.168.1.5
;
; Create an entry for the bootstrap host.
bootstrap.ocp4 IN A 192.168.1.96
;
; Create entries for the master hosts.
master0.ocp4  IN A 192.168.1.97
```

```
master1.ocp4  IN A 192.168.1.98
master2.ocp4  IN A 192.168.1.99
;
; Create entries for the worker hosts.
worker0.ocp4  IN A 192.168.1.11
worker1.ocp4  IN A 192.168.1.7
;
;EOF
```

下面的 BIND 区文件示例显示了反向名字解析的 PTR 记录示例。

例 5.2. 反向记录的 DNS 区数据库示例

```
$TTL 1W
@ IN SOA ns1.example.com. root (
   2019070700 ; serial
   3H  ; refresh (3 hours)
   30M  ; retry (30 minutes)
   2W  ; expiry (2 weeks)
   1W )  ; minimum (1 week)
 IN NS ns1.example.com.
;
; The syntax is "last octet" and the host must have an FQDN
; with a trailing dot.
97 IN PTR master0.ocp4.example.com.
98 IN PTR master1.ocp4.example.com.
99 IN PTR master2.ocp4.example.com.
;
96 IN PTR bootstrap.ocp4.example.com.
;
5 IN PTR api.ocp4.example.com.
5 IN PTR api-int.ocp4.example.com.
;
11 IN PTR worker0.ocp4.example.com.
7 IN PTR worker1.ocp4.example.com.
;
;EOF
```

5.1.5. 生成 SSH 私钥并将其添加到代理中

如果要在集群上执行安装调试或灾难恢复，则必须为 **ssh-agent** 和安装程序提供 SSH 密钥。您可以使用此密钥访问公共集群中的 bootstrap 机器来排除安装问题。

注意

在生产环境中，您需要进行灾难恢复和调试。

您可以使用此密钥以 **core** 用户身份通过 SSH 连接到 master 节点。在部署集群时，此密钥会添加到 **core** 用户的 **~/.ssh/authorized_keys** 列表中。

注意

您必须使用一个本地密钥，而不要使用在特定平台上配置的密钥，如 AWS 密钥对。

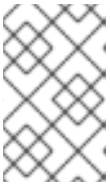
流程

1. 如果还没有为计算机上免密码身份验证而配置的 SSH 密钥，请创建一个。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ ssh-keygen -t ed25519 -N '' \
  -f <path>/<file_name> 1
```

1 指定新 SSH 密钥的路径和文件名，如 **~/.ssh/id_rsa**。如果您已有密钥对，请确保您的公钥位于 **~/.ssh** 目录中。

运行此命令会在指定的位置生成不需要密码的 SSH 密钥。



注意

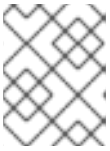
如果您计划在 **x86_64** 架构中安装使用 FIPS 验证的/Modules in Process 加密库的 OpenShift Container Platform 集群，不要创建使用 **ed25519** 算法的密钥。反之，创建一个使用 **rsa** 或 **ecdsa** 算法的密钥。

2. 作为后台任务启动 **ssh-agent** 进程：

```
$ eval "$(ssh-agent -s)"
```

输出示例

```
Agent pid 31874
```



注意

如果您的集群采用 FIPS 模式，则只使用 FIPS 兼容算法来生成 SSH 密钥。密钥必须是 RSA 或 ECDSA。

3. 将 SSH 私钥添加到 **ssh-agent**：

```
$ ssh-add <path>/<file_name> 1
```

输出示例

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

1 指定 SSH 私钥的路径和文件名，如 **~/.ssh/id_rsa**

后续步骤

- 在安装 OpenShift Container Platform 时，为安装程序提供 SSH 公钥。如果在您置备的基础架构上安装集群，您必须将此密钥提供给集群的机器。

5.1.6. 获取安装程序

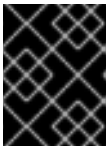
在安装 OpenShift Container Platform 之前，将安装文件下载到本地计算机上。

先决条件

- 运行 Linux 或 macOS 的计算机，本地磁盘空间为 500 MB

流程

1. 访问 OpenShift Cluster Manager 站点的 Infrastructure Provider 页面。如果您有红帽帐号，请使用自己的凭证登录。如果没有，请创建一个帐户。
2. 选择您的基础架构供应商。
3. 进入适用于您的安装类型的页面，下载您的操作系统的安装程序，并将文件放在要保存安装配置文件的目录中。。



重要

安装程序会在用来安装集群的计算机上创建若干文件。在完成集群安装后，您必须保留安装程序和安装程序所创建的文件。这两个文件都需要删除集群。

重要

删除安装程序创建的文件不会删除您的集群，即使集群在安装过程中失败也是如此。要删除集群，为特定云供应商完成 OpenShift Container Platform 卸载流程。

4. 提取安装程序。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ tar xvf openshift-install-linux.tar.gz
```

5. 从 Red Hat OpenShift Cluster Manager 下载安装 pull secret 。通过此 pull secret，您可以进行所含授权机构提供的服务的身份验证，这些服务包括为 OpenShift Container Platform 组件提供容器镜像的 Quay.io。

5.1.7. 通过下载二进制文件安装 OpenShift CLI

您需要安装 CLI（**oc**） 来使用命令行界面与 OpenShift Container Platform 进行交互。您可在 Linux、Windows 或 macOS 上安装 **oc**。

重要

如果安装了旧版本的 **oc**，则无法使用 OpenShift Container Platform 4.6 中的所有命令。下载并安装新版本的 **oc**。

5.1.7.1. 在 Linux 上安装 OpenShift CLI

您可以按照以下流程在 Linux 上安装 OpenShift CLI（**oc**）二进制文件。

流程

1. 进入到红帽客户门户网站上的 OpenShift Container Platform 下载页面。

2. 在 **Version** 下拉菜单中选择相应的版本。
3. 单击 **OpenShift v4.6 Linux 客户端** 条目旁边的 **Download Now**，再保存文件。
4. 解包存档：

```
$ tar xvzf <file>
```

5. 把 **oc** 二进制代码放到 **PATH** 中的目录中。
   执行以下命令可以查看当前的 **PATH** 设置：

```
$ echo $PATH
```

安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
$ oc <command>
```

5.1.7.2. 在 Windows 上安装 OpenShift CLI

您可以按照以下流程在 Windows 上安装 OpenShift CLI（**oc**）二进制代码。

流程

1. 进入到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 在 **Version** 下拉菜单中选择相应的版本。
3. 单击 **OpenShift v4.6 Windows 客户端**条目旁边的 **Download Now**，再保存文件。
4. 使用 ZIP 程序解压存档。
5. 把 **oc** 二进制代码放到 **PATH** 中的目录中。
   要查看您的 **PATH**，请打开命令提示窗口并执行以下命令：

```
C:\> path
```

安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
C:\> oc <command>
```

5.1.7.3. 在 macOS 上安装 OpenShift CLI

您可以按照以下流程在 macOS 上安装 OpenShift CLI（**oc**）二进制代码。

流程

1. 进入到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 在 **Version** 下拉菜单中选择相应的版本。
3. 单击 **OpenShift v4.6 MacOSX 客户端**条目旁边的 **Download Now**，再保存文件。
4. 解包和解压存档。

5. 将 **oc** 二进制文件移到 PATH 的目录中。
   要查看您的 **PATH**，打开一个终端窗口并执行以下命令：

```
$ echo $PATH
```

安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
$ oc <command>
```

5.1.8. 手动创建安装配置文件

对于使用用户置备的基础架构的 OpenShift Container Platform 安装，您必须手动生成安装配置文件。

先决条件

- 获取 OpenShift Container Platform 安装程序和集群的访问令牌。

流程

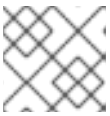
1. 创建用来存储您所需的安装资产的安装目录：

```
$ mkdir <installation_directory>
```

重要

您必须创建目录。一些安装信息，如 bootstrap X.509 证书，有较短的过期间隔，因此不要重复使用安装目录。如果要重复使用另一个集群安装中的个别文件，可以将其复制到您的目录中。但是，一些安装数据的文件名可能会在发行版本之间有所改变。从 OpenShift Container Platform 老版本中复制安装文件时要格外小心。

2. 自定义以下 **install-config.yaml** 文件模板，并将它保存到 **<installation_directory>** 中。



注意

此配置文件必须命名为 **install-config.yaml**。

3. 备份 **install-config.yaml** 文件，以便用于安装多个集群。

重要

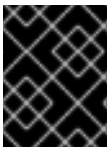
install-config.yaml 文件会在安装过程的下一步骤中消耗掉。现在必须备份它。

5.1.8.1. 安装配置参数

在部署 OpenShift Container Platform 集群前，您可以提供参数值，以描述托管集群的云平台的帐户并选择性地自定义集群平台。在创建 **install-config.yaml** 安装配置文件时，您可以通过命令行来提供所需的参数的值。如果要自定义集群，可以修改 **install-config.yaml** 文件来提供关于平台的更多信息。

注意

安装之后，您无法修改 **install-config.yaml** 文件中的这些参数。



重要

openshift-install 命令不验证参数的字段名称。如果指定了不正确的名称，则不会创建相关的文件或对象，且不会报告错误。确保所有指定的参数的字段名称都正确。

5.1.8.1.1. 所需的配置参数

下表描述了所需的安装配置参数：

表 5.8. 所需的参数

参数	描述	值
apiVersion	**install-config.yaml** 内容的 API 版本。当前版本是 **v1**。安装程序还可能支持旧的 API 版本。	字符串
baseDomain	云供应商的基域。此基础域用于创建到 OpenShift Container Platform 集群组件的路由。集群的完整 DNS 名称是 **baseDomain** 和 **metadata.name** 参数值的组合，其格式为 **<metadata.name>.<baseDomain>**。	完全限定域名或子域名，如 **example.com**。
metadata	Kubernetes 资源 **ObjectMeta**，其中只消耗 **name** 参数。	对象
metadata.name	集群的名称。集群的 DNS 记录是 **{{.metadata.name}}.{{.baseDomain}}** 的子域。	小写字母,连字符(-)和句点(.)的字符串，如 **dev**。
platform	执行安装的具体平台配置：**aws**、**baremetal**、**azure**、**openstack**、**ovirt**、**vsphere**。有关 **platform.<platform>** 参数的额外信息，请参考下表来了解您的具体平台。	对象

参数	描述	值

pullSecret	从 Red Hat OpenShift Cluster Manager 获取 pull secret，验证从 Quay.io 等服务中下载 OpenShift Container Platform 组件的容器镜像。	<pre>{
		  "auths":{
		  "cloud.openshift.com":{
		  "auth":"b3Blb=",
		  "email":"you@example.com"
		  },
		  "quay.io":{
		  "auth":"b3Blb=",
		  "email":"you@example.com"
		  }
		  }
		}</pre>

5.1.8.1.2. 网络配置参数

您可以根据现有网络基础架构的要求自定义安装配置。例如，您可以扩展集群网络的 IP 地址块，或者提供不同于默认值的不同 IP 地址块。

只支持 IPv4 地址。

表 5.9. 网络参数

参数	描述	值
networking	集群网络的配置。	对象 **注意** 您不能在安装后修改 **networking** 对象指定的参数。
networking.networkType	要安装的集群网络供应商 Container Network Interface（CNI）插件。	**OpenShiftSDN** 或 **OVNKubernetes**。默认值为 **OpenShiftSDN**。

networking.clusterNetwork	pod 的 IP 地址块。
	
	默认值为 **10.128.0.0/14**，主机前缀为 **/23**。
	
	如果您指定多个 IP 地址块，则块不得互相重叠。
		一个对象数组。例如：
		
		<pre>networking:
		  clusterNetwork:
		  - cidr: 10.128.0.0/14
		  hostPrefix: 23</pre>

networking.clusterNetwork.cidr	使用 **networking.clusterNetwork** 时需要此项。IP 地址块。 一个 IPv4 网络。	使用 CIDR 形式的 IP 地址块。IPv4 块的前缀长度介于 **0** 到 **32** 之间。

参数	描述	值
networking.clusterNetwork.hostPrefix	分配给每个单独节点的子网前缀长度。例如，如果 **hostPrefix** 设为 **23**，则每个节点从所给的 **cidr** 中分配一个 **/23** 子网。**hostPrefix** 值 **23** 提供 510（2^(32 - 23)- 2）个 pod IP 地址。	子网前缀。 默认值为 **23**。
networking.serviceNetwork	服务的 IP 地址块。默认值为 **172.30.0.0/16**。 OpenShift SDN 和 OVN-Kubernetes 网络供应商只支持服务网络的一个 IP 地址块。	CIDR 格式具有 IP 地址块的数组。例如： networking: serviceNetwork: - 172.30.0.0/16
networking.machineNetwork	机器的 IP 地址块。 如果您指定多个 IP 地址块，则块不得互相重叠。	一个对象数组。例如： networking: machineNetwork: - cidr: 10.0.0.0/16
networking.machineNetwork.cidr	使用 **networking.machineNetwork** 时需要。IP 地址块。libvirt 以外的所有平台的默认值为 **10.0.0.0/16**。对于 libvirt，默认值为 **192.168.126.0/24**。	CIDR 表示法中的 IP 网络块。 例如： **10.0.0.0/16**。 **注意** 将 **networking.machineNetwork** 设置为与首选 NIC 所在的 CIDR 匹配。

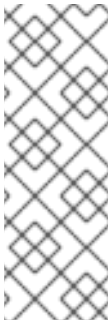
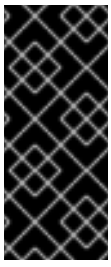
5.1.8.1.3. 可选配置参数

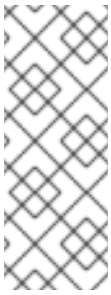
下表描述了可选安装配置参数：

表 5.10. 可选参数

参数	描述	值
additionalTrustBundle	添加到节点可信证书存储中的 PEM 编码 X.509 证书捆绑包。配置了代理时，也可以使用这个信任捆绑包。	字符串
compute	组成计算节点的机器的配置。	machine-pool 对象的数组。详情请查看以下"Machine-pool"表。

参数	描述	值
compute.architecture	决定池中机器的指令集合架构。目前不支持异构集群，因此所有池都必须指定相同的架构。有效值为 **amd64** （默认值）。	字符串
compute.hyperthreading	是否在计算机器上启用或禁用并发多线程或**超线程**。默认情况下，启用并发多线程以提高机器内核的性能。  **重要** 如果禁用并发多线程，请确保在容量规划时考虑到机器性能可能会显著降低的问题。	**Enabled** 或 **Disabled**
compute.name	使用 **compute** 时需要此值。机器池的名称。	**worker**
compute.platform	使用 **compute** 时需要此值。使用此参数指定托管 worker 机器的云供应商。此参数值必须与 **controlPlane.platform** 参数值匹配。	**aws**、**azure**、**gcp**、**openstack**、**ovirt**、**vsphere** 或 **{}**
compute.replicas	要置备的计算机器数量，也称为 worker 机器。	大于或等于 **2** 的正整数。默认值为 **3**。
controlPlane	组成 control plane 的机器的配置。	**MachinePool** 对象的数组。详情请查看以下"Machine-pool"表。
controlPlane.architecture	决定池中机器的指令集合架构。目前不支持异构集群，因此所有池都必须指定相同的架构。有效值为 **amd64** （默认值）。	字符串
controlPlane.hyperthreading	是否在 control plane 机器上启用或禁用并发多线程或**超线程**。默认情况下，启用并发多线程以提高机器内核的性能。  **重要** 如果禁用并发多线程，请确保在容量规划时考虑到机器性能可能会显著降低的问题。	**Enabled** 或 **Disabled**

参数	描述	值
controlPlane.name	使用 **controlPlane** 时需要。机器池的名称。	**master**
controlPlane.platform	使用 **controlPlane** 时需要。使用此参数指定托管 control plane 机器的云供应商。此参数值必须与 **compute.platform** 参数值匹配。	**aws**、**azure**、**gcp**、**openstack**、**ovirt**、**vsphere** 或 **{}**
controlPlane.replicas	要置备的 control plane 机器数量。	唯一支持的值是 **3**，它是默认值。
credentialsMode	Cloud Credential Operator（CCO）模式。如果没有指定任何模式，CCO 会动态地尝试决定提供的凭证的功能，在支持多个模式的平台上使用 mint 模式。  **注意** 不是所有 CCO 模式都支持所有云供应商。如需有关 CCO 模式的更多信息，请参阅*Red Hat Operator 参考指南*内容中的 *Cloud Credential Operator* 条目。	**Mint**、**Passthrough**、**Manual** 或空字符串(**""**)。
fips	启用或禁用 FIPS 模式。默认为 **false**（禁用）。如果启用了 FIPS 模式，运行 OpenShift Container Platform 的 Red Hat Enterprise Linux CoreOS (RHCOS) 机器会绕过默认的 Kubernetes 加密套件，并使用由 RHCOS 提供的加密模块。  **重要** 只有在 **x86_64** 架构中的 OpenShift Container Platform 部署支持 FIPS 验证的/Modules in Process 加密库。  **注意** 如果使用 Azure File 存储，则无法启用 FIPS 模式。	**false** 或 **true**

参数	描述	值
imageContentSources	release-image 内容的源和仓库。	对象数组。包括一个 **source** 以及可选的 **mirrors**，如下表所示。
imageContentSources.source	使用 **imageContentSources** 时需要。指定用户在镜像拉取规格中引用的仓库。	字符串
imageContentSources.mirrors	指定可能还包含同一镜像的一个或多个仓库。	字符串数组
publish	如何发布或公开集群的面向用户的端点，如 Kubernetes API、OpenShift 路由。	**Internal** 或 **External**。默认值为 **External**。 在非云平台上不支持将此字段设置为 **Internal**。  **重要** 如果将字段的值设为 **Internal**，集群将无法运行。如需更多信息，请参阅 BZ#1953035。
sshKey	用于验证集群机器访问的 SSH 密钥或密钥。  **注意** 对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 **ssh-agent** 进程使用的 SSH 密钥。	一个或多个密钥。例如： sshKey: <key1> <key2> <key3>

5.1.8.2. 裸机 install-config.yaml 文件示例

您可以自定义 **install-config.yaml** 文件，以指定有关 OpenShift Container Platform 集群平台的更多信息，或修改所需参数的值。

```
apiVersion: v1
baseDomain: example.com (1)
compute: (2)
- hyperthreading: Enabled (3)
  name: worker
  replicas: 0 (4)
controlPlane: (5)
  hyperthreading: Enabled (6)
```

```
  name: master
  replicas: 3 7
metadata:
  name: test 8
networking:
  clusterNetwork:
  - cidr: 10.128.0.0/14 9
    hostPrefix: 23 10
  networkType: OpenShiftSDN
  serviceNetwork: 11
  - 172.30.0.0/16
platform:
  none: {} 12
fips: false 13
pullSecret: '{"auths": ...}' 14
sshKey: 'ssh-ed25519 AAAA...' 15
```

1 集群的基域。所有 DNS 记录都必须是这个基域的子域，并包含集群名称。

2 5 **controlPlane** 部分是一个单映射，但 **compute** 部分是一系列映射。为满足不同数据结构的要求，**compute** 部分的第一行必须以连字符 **-** 开头，**controlPlane** 部分的第一行则不可以连字符开头。只使用一个 control plane 池。

3 6 是否要启用或禁用并发多线程（SMT）或**超线程**。默认情况下，启用 SMT 可提高机器内核的性能。您可以通过将参数值设为 **Disabled** 来禁用。如果禁用 SMT，则必须在所有集群机器中禁用它，其中包括 control plane 和计算机器。

注意

默认启用并发多线程（SMT）。如果在 BIOS 设置中没有启用 SMT，**hyperthreading** 参数不会起作用。

重要

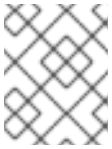
如果您禁用 **hyperthreading**（无论是在 BIOS 中还是在 **install-config.yaml** 中），请确保您对可能会造成的机器性能显著降低的情况有所考虑。

4 **replicas** 参数的值必须设置为 **0**。此参数控制集群为您创建和管理的 worker 数量，使用用户置备的基础架构时集群不会执行这些功能。在完成 OpenShift Container Platform 安装前，您必须手动为集群部署 worker 机器。

7 您添加到集群的 control plane 机器数量。由于集群将这个值用作集群中 etcd 端点的数量，因此该值必须与您部署的 control plane 机器数量匹配。

8 您在 DNS 记录中指定的集群名称。

9 从中分配 pod IP 地址的 IP 地址块。此块不得与现有的物理网络重叠。这些 IP 地址用于 pod 网络。如果您需要从外部网络访问 pod，请配置负载均衡器和路由器来管理流量。



注意

类 E CIDR 范围保留给以后使用。要使用 Class E CIDR 范围，您必须确保您的网络环境接受 Class E CIDR 范围内的 IP 地址。

10 分配给每个单独节点的子网前缀长度。例如，如果 **hostPrefix** 设为 **23**，则每个节点从所给的 **cidr** 中分配一个 **/23** 子网，这样就能有 510 (2^(32 - 23) - 2) 个 Pod IP 地址。如果您需要从外部网络访

11 用于服务 IP 地址的 IP 地址池。您只能输入一个 IP 地址池。此块不得与现有的物理网络重叠。如果您需要从外部网络访问服务，请配置负载均衡器和路由器来管理流量。

12 您必须将平台设置为 **none**。您不能为您的平台提供额外的平台配置变量。

13 是否启用或禁用 FIPS 模式。默认情况下不启用 FIPS 模式。如果启用了 FIPS 模式，运行 OpenShift Container Platform 的 Red Hat Enterprise Linux CoreOS (RHCOS) 机器会绕过默认的 Kubernetes 加密套件，并使用由 RHCOS 提供的加密模块。

重要

只有在 **x86_64** 架构中的 OpenShift Container Platform 部署支持 FIPS 验证的/Modules in Process 加密库。

14 Red Hat OpenShift Cluster Manager 中的 pull secret 。通过此 pull secret，您可以进行所含授权机构提供的服务的身份验证，这些服务包括为 OpenShift Container Platform 组件提供容器镜像的 Quay.io。

15 Red Hat Enterprise Linux CoreOS (RHCOS) 中 **core** 用户的默认 SSH 密钥的公钥部分。

注意

对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 **ssh-agent** 进程使用的 SSH 密钥。

5.1.8.3. 在安装过程中配置集群范围代理

生产环境可能会拒绝直接访问互联网，而是提供 HTTP 或 HTTPS 代理。您可以通过在 **install-config.yaml** 文件中配置代理设置，将新的 OpenShift Container Platform 集群配置为使用代理。

注意

对于裸机安装，如果您没有从 **install-config.yaml** 文件中的 **networking.machineNetwork[].cidr** 字段指定的范围分配节点 IP 地址，您必须将其包括在 **proxy.noProxy** 字段中。

先决条件

- 您有一个现有的 **install-config.yaml** 文件。
- 您检查了集群需要访问的站点，并决定是否需要绕过代理。默认情况下代理所有集群出口流量，包括对托管云供应商 API 的调用。您需要将站点添加到 **Proxy** 对象的 **spec.noProxy** 字段来绕过代理。

注意

Proxy 对象 **status.noProxy** 字段使用安装配置中的 **networking.machineNetwork[].cidr**、**networking.clusterNetwork[].cidr** 和 **networking.serviceNetwork[]** 字段的值填充。

对于在 Amazon Web Services(AWS)、Google Cloud Platform(GCP)、Microsoft Azure 和 Red Hat OpenStack Platform(RHOSP)上安装, **Proxy** 对象 **status.noProxy** 字段也会使用实例元数据端点填充(**169.254.169.254**)。

流程

1. 编辑 **install-config.yaml** 文件并添加代理设置。例如：

```
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: example.com 3
additionalTrustBundle: | 4
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
...
```

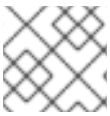
1. 用于创建集群外 HTTP 连接的代理 URL。URL 必须是 **http**。
2. 用于创建集群外 HTTPS 连接的代理 URL。
3. 要排除在代理中的目标域名、IP 地址或其他网络 CIDR 的逗号分隔列表。在域前面加 **.** 来仅匹配子域。例如： **.y.com** 匹配 **x.y.com**，但不匹配 **y.com**。使用 * 绕过所有目的地的代理。
4. 如果提供，安装程序会在 **openshift-config 命名空间中生成名为 user-ca- bundle** 的配置映射来保存额外的 CA 证书。如果您提供 **additionalTrustBundle** 和至少一个代理设置，则 **Proxy** 对象会被配置为引用 **trustedCA** 字段中的 **user-ca-bundle** 配置映射。然后，Cluster Network Operator 会创建一个 **trusted-ca-bundle** 配置映射，该配置映射将为 **trustedCA** 参数指定的内容与 RHCOS 信任捆绑包合并。 **additionalTrustBundle** 字段是必需的，除非代理的身份证书由来自 RHCOS 信任捆绑包的颁发机构签名。

注意

安装程序不支持代理的 **readinessEndpoints** 字段。

2. 保存该文件，并在安装 OpenShift Container Platform 时引用。

安装程序会创建一个名为 **cluster** 的集群范围代理，该代理使用提供的 **install-config.yaml** 文件中的代理设置。如果没有提供代理设置，仍然会创建一个 **cluster Proxy** 对象，但它会有一个空 **spec**。



注意

只支持名为 **cluster** 的 **Proxy** 对象，且无法创建额外的代理。

5.1.9. 配置三节点集群

您可在没有 worker 的 OpenShift Container Platform 中安装和运行三节点集群。这为集群管理员和开发人员提供了较小的、效率更高的集群，用于开发、生产及测试。

流程

- 编辑 **install-config.yaml** 文件，将计算副本（也称为 worker 副本）数设为 **0**，如以下 **compute** 小节中所示：

```
compute:
- name: worker
  platform: {}
  replicas: 0
```

5.1.10. 创建 Kubernetes 清单和 Ignition 配置文件

由于您必须修改一些集群定义文件并要手动启动集群机器，因此您必须生成 Kubernetes 清单和 Ignition 配置文件，集群需要这两项来创建其机器。

安装配置文件转换为 Kubernetes 清单。清单嵌套到 Ignition 配置文件中，稍后用于创建集群。

重要

- 安装程序生成的 Ignition 配置文件包含在 24 小时后过期的证书，然后在过期时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外情况是，您需要手动批准待处理的 **node-bootstrapper** 证书签名请求（CSR）来恢复 kubelet 证书。如需更多信息，请参阅*从过期的 control plane 证书中恢复*的文档。
- 建议您在生成 12 小时后使用 Ignition 配置文件，因为集群安装后 24 小时证书从 16 小时轮转至 22 小时。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中运行证书更新时避免安装失败。

先决条件

- 已获得 OpenShift Container Platform 安装程序。
- 已创建 **install-config.yaml** 安装配置文件。

流程

1. 切换到包含安装程序的目录，并为集群生成 Kubernetes 清单：

```
$ ./openshift-install create manifests --dir <installation_directory> 1
```

1 对于 **<installation_directory>**，请指定含有您创建的 **install-config.yaml** 文件的安装目录。

警告

如果要安装一个三节点集群，请跳过以下步骤，以便 control plane 节点可以调度。

+

重要

当您将 control plane 节点从默认的不可调度配置为可以调度时，需要额外的订阅。这是因为 control plane 节点随后变为 worker 节点。

1. 检查 **<installation_directory>/manifests/cluster-scheduler-02-config.yml** Kubernetes 清单文件中的 **mastersSchedulable** 参数是否已设置为 **false**。此设置可防止在 control plane 机器上调度 pod:
   a. 打开 **<installation_directory>/manifests/cluster-scheduler-02-config.yml** 文件。
   b. 找到 **mastersSchedulable** 参数并确保它被设置为 **false**。
   c. 保存并退出文件。
2. 要创建 Ignition 配置文件，从包含安装程序的目录运行以下命令：

```
$ ./openshift-install create ignition-configs --dir <installation_directory> 1
```

1 对于 **<installation_directory>**，请指定相同的安装目录。

该目录中将生成以下文件：

```
.
├── auth
│   ├── kubeadmin-password
│   └── kubeconfig
├── bootstrap.ign
├── master.ign
├── metadata.json
└── worker.ign
```

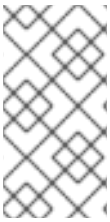
其他资源

- 如需有关 恢复 kubelet 证书的更多信息，请参阅恢复已过期的 control plane 证书。

5.1.11. 安装 RHCOS 并启动 OpenShift Container Platform bootstrap 过程

要在您置备的裸机基础架构上安装 OpenShift Container Platform，您必须在机器上安装 Red Hat Enterprise Linux CoreOS（RHCOS）。安装 RHCOS 时，您必须为 OpenShift Container Platform 安装程序生成的机器类型提供 Ignition 配置文件。如果您配置了合适的网络、DNS 和负载均衡基础架构，OpenShift Container Platform bootstrap 过程会在 RHCOS 机器重启后自动开始。

要在机器上安装 RHCOS，请按照以下步骤使用 ISO 镜像或网络 PXE 启动。



注意

本安装文档中包括的计算节点部署步骤特定于 RHCOS。如果您选择部署基于 RHEL 的计算节点，您将接管所有操作系统生命周期管理和维护，包括执行系统更新、应用补丁和完成所有其他必要的任务。RHEL 7 计算机器的使用已弃用，计划在以后的 OpenShift Container Platform 4 发行版本中删除。

您可以使用以下方法在 ISO 和 PXE 安装过程中配置 RHCOS:

- 内核参数： 您可以使用内核参数来提供特定于安装的信息。例如，您可以指定上传到 HTTP 服务器的 RHCOS 安装文件的位置，以及您要安装的节点类型的 Ignition 配置文件的位置。对于 PXE 安装，您可以使用 **APPEND** 参数将参数传递给实时安装程序的内核。对于 ISO 安装，您可以中断实时安装引导过程来添加内核参数。在这两种安装情况下，您可以使用特殊的 **coreos.inst.*** 参数来指示实时安装程序，以及标准安装引导参数来打开或关闭标准内核服务。

- Ignition 配置：OpenShift Container Platform Ignition 配置文件（***.ign**）特定于您要安装的节点类型。您可以在 RHCOS 安装过程中传递 bootstrap、control plane 或计算节点 Ignition 配置文件的位置，以便在第一次引导时生效。特殊情况下，您可以创建单独的、有限的 Ignition 配置来传递给 Live 系统。该 Ignition 配置可以执行特定任务，如在安装完成后向置备系统报告成功。这个特殊 Ignition 配置由 **coreos-installer** 使用，用于首次启动安装的系统。不要直接向 live ISO 提供标准 control plane 和计算节点 Ignition 配置。

- **coreos-installer**：您可以将 live ISO 安装程序引导到 shell 提示符，这可让您在首次引导前以多种方式准备持久性系统。特别是，您可以运行 **coreos-installer** 命令来识别包括的工件、使用磁盘分区以及设置联网。在有些情况下，您可以配置 live 系统上的功能并将其复制到安装的系统中。

使用 ISO 安装还是 PXE 安装要根据您的具体情况而定。PXE 安装需要可用的 DHCP 服务并进行更多准备，但可以使安装过程更自动化。ISO 安装是一个更手动的过程，如果您设置的机器较多，则可能不方便。

注意

自 OpenShift Container Platform 4.6 起，RHCOS ISO 和其他安装工件支持在带有 4K 扇区的磁盘上安装。

5.1.11.1. 使用 ISO 镜像创建 Red Hat Enterprise Linux CoreOS (RHCOS) 机器

在您置备的基础架构上安装集群前，必须先创建 RHCOS 机器供其使用。您可以使用 ISO 镜像来创建这些机器。

先决条件

- 获取集群的 Ignition 配置文件。
- 具有可从计算机以及您创建的机器访问的 HTTP 服务器的访问权限。

流程

1. 将安装程序创建的 control plane、计算和 bootstrap Ignition 配置文件上传到 HTTP 服务器。记下这些文件的 URL。

重要

如果您计划在安装完成后在集群中添加更多计算机器，请不要删除这些文件。

2. 从 RHCOS 镜像镜像 页面获取您选择的操作系统实例安装方法所需的 RHCOS 镜像。

重要

RHCOS 镜像可能不会随着 OpenShift Container Platform 的每一发行版本都有改变。您必须下载最高版本的镜像，其版本号应小于或等于您安装的 OpenShift Container Platform 版本。如果可用，请使用与 OpenShift Container Platform 版本匹配的镜像版本。此流程只使用 ISO 镜像。此安装类型不支持 RHCOS qcow2 镜像。

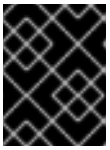
ISO 文件名类似以下示例：

rhcos-<version>-live.<architecture>.iso

3. 使用 ISO 启动 RHCOS 安装。使用如下安装选项之一：
   - 将 ISO 镜像刻录到磁盘并直接启动。
   - 通过 LOM 接口使用 ISO 重定向。
4. 引导 ISO 镜像。您可以中断安装引导过程来添加内核参数。然而，在这个 ISO 过程中，您应该使用 **coreos-installer** 命令而不是添加内核参数。如果您在没有选项或中断的情况下运行 live 安装程序，安装程序将引导至 live 系统上的 shell 提示符，准备好将 RHCOS 安装到磁盘中。
5. 在运行 **coreos-installer** 前，请参阅 *高级 RHCOS 安装参考* 部分，以了解配置功能的不同方法，如网络和磁盘分区。
6. 运行 **coreos-installer** 命令。您至少必须识别节点类型的 Ignition 配置文件位置，以及您要安装到的磁盘位置。下面是一个示例：

```
$ sudo coreos-installer install \
    --ignition-url=https://host/worker.ign /dev/sda
```

7. 安装 RHCOS 后，系统会重启。系统重启过程中，它会应用您指定的 Ignition 配置文件。
8. 继续为集群创建其他机器。



重要

此刻您必须创建 bootstrap 和 control plane 机器。如果 control plane 机器不可调度（这是默认调度），则在安装集群前至少会创建两台计算机器。

5.1.11.2. 通过 PXE 或 iPXE 启动来创建 Red Hat Enterprise Linux CoreOS (RHCOS) 机器

在安装使用手动置备 RHCOS 节点（如裸机）的集群前，您必须创建 RHCOS 机器供其使用。您可以使用 PXE 或 iPXE 启动来创建机器。

先决条件

- 获取集群的 Ignition 配置文件。

- 配置合适的 PXE 或 iPXE 基础架构。
- 具有 HTTP 服务器的访问权限，以便您可从计算机进行访问。

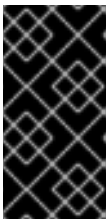
流程

1. 将安装程序创建的 master、worker 和 bootstrap Ignition 配置文件上传到 HTTP 服务器。记下这些文件的 URL。

重要

您可以在 Ignition 配置中添加或更改配置设置，然后将其保存到 HTTP 服务器。如果您计划在安装完成后在集群中添加更多计算机器，请不要删除这些文件。

2. 从 RHCOS 镜像 镜像页面获取 RHCOS **内核、initram fs 和 rootfs** 文件。



重要

RHCOS 工件（artifact）可能不会随着 OpenShift Container Platform 的每个发行版本而改变。您必须下载最高版本的工件，其版本号应小于或等于您安装的 OpenShift Container Platform 版本。这个过程只使用下面描述的正确 **kernel**、**initramfs** 和 **rootfs** 工件。此安装类型不支持 RHCOS qcow2 镜像。

文件名包含 OpenShift Container Platform 版本号。它们类似以下示例：

- **kernel**: **rhcos-<version>-live-kernel-<architecture>**
- **initramfs**: **rhcos-<version>-live-initramfs.<architecture>.img**
- **rootfs**: **rhcos-<version>-live-rootfs.<architecture>.img**

3. 上传引导方法所需的额外文件：
   - 对于传统的 PXE，将 **kernel** 和 **initramfs** 文件上传到 TFTP 服务器，并将 **rootfs** 文件上传到 HTTP 服务器。
   - 对于 iPXE，将 **kernel**、**initram** fs 和 **rootfs** 文件上传到 HTTP 服务器。

重要

如果您计划在安装完成后在集群中添加更多计算机器，请不要删除这些文件。

4. 配置网络启动基础架构，以便在安装 RHCOS 后机器可从本地磁盘启动。
5. 为 RHCOS 镜像配置 PXE 或 iPXE 安装。
   针对您的环境修改以下示例菜单条目之一，并验证能否正确访问镜像和 Ignition 文件：
   - 对于 PXE：

```
DEFAULT pxeboot
TIMEOUT 20
PROMPT 0
LABEL pxeboot
    KERNEL http://<HTTP_server>/rhcos-<version>-live-kernel-<architecture> (1)
```

```
    APPEND initrd=http://<HTTP_server>/rhcos-<version>-live-initramfs.
<architecture>.img coreos.live.rootfs_url=http://<HTTP_server>/rhcos-<version>-live-
rootfs.<architecture>.img coreos.inst.install_dev=/dev/sda
coreos.inst.ignition_url=http://<HTTP_server>/bootstrap.ign 2 3
```

1. 指定上传到 HTTP 服务器的 live **kernel** 文件位置。URL 必须是 HTTP、TFTP 或者 FTP ; 不支持 HTTPS 和 NFS。
2. 如果您使用多个 NIC，请在 **ip** 选项中指定一个接口。例如，要在名为 **eno1** 的 NIC 上使用 DHCP，请设置 **ip=eno1:dhcp**。
3. 指定上传到 HTTP 服务器的 RHCOS 文件的位置。**initrd** 参数值是 **initramfs** 文件的位置，**coreos.live.rootfs_url** 参数值是 **rootfs** 文件的位置，**coreos.inst.ignition_url** 参数值是 bootstrap Ignition 配置文件的位置。您还可以在 **APPEND** 行中添加更多内核参数来配置联网或其他引导选项。

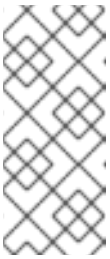
注意

这个配置不会在使用图形控制台的机器上启用串口控制台访问。要配置不同的控制台，请在 **APPEND** 行中添加一个或多个 **console=** 参数。例如，添加 **console=tty0 console=ttyS0** 将第一个 PC 串口设置为主控制台，图形控制台作为二级控制台。如需更多信息，请参阅如何在 Red Hat Enterprise Linux 中设置串行终端和（或）控制台？

- 对于 iPXE：

```
kernel http://<HTTP_server>/rhcos-<version>-live-kernel-<architecture> initrd=main
coreos.live.rootfs_url=http://<HTTP_server>/rhcos-<version>-live-rootfs.
<architecture>.img coreos.inst.install_dev=/dev/sda
coreos.inst.ignition_url=http://<HTTP_server>/bootstrap.ign 1 2
initrd --name main http://<HTTP_server>/rhcos-<version>-live-initramfs.
<architecture>.img 3
boot
```

1. 指定上传到 HTTP 服务器的 RHCOS 文件的位置。**kernel** 参数值是 **kernel** 文件的位置，在 UEFI 系统中引导时需要 **initrd=main** 参数。 **coreos.live.rootfs_url** 参数值是 **rootfs** 文件的位置， **coreos.inst.ignition_url** 参数值则是 bootstrap Ignition 配置文件的位置。
2. 如果您使用多个 NIC，请在 **ip** 选项中指定一个接口。例如，要在名为 **eno1** 的 NIC 上使用 DHCP，请设置 **ip=eno1:dhcp**。
3. 指定上传到 HTTP 服务器的 **initramfs** 文件的位置。



注意

这个配置不会在使用图形控制台的机器上启用串口控制台访问。要配置不同的控制台，请在 **kerne** 行中添加一个或多个 **console=** 参数。例如，添加 **console=tty0 console=ttyS0** 将第一个 PC 串口设置为主控制台，图形控制台作为二级控制台。如需更多信息，请参阅如何在 Red Hat Enterprise Linux 中设置串行终端和（或）控制台？

6. 如果使用 PXE UEFI，请执行以下操作：

   a. 提供引导系统所需的 **shim x64.efi 和 grubx64 .efi EFI 二进制文件以及 grub.cfg** 文件。

      - 通过将 RHCOS ISO 挂载到主机，然后将 **images/efiboot.img** 文件挂载到您的主机来提取所需的 EFI 二进制文件：

        ```
        $ mkdir -p /mnt/iso
        ```

        ```
        $ mkdir -p /mnt/efiboot
        ```

        ```
        $ mount -o loop rhcos-installer.x86_64.iso /mnt/iso
        ```

        ```
        $ mount -o loop,ro /mnt/iso/images/efiboot.img /mnt/efiboot
        ```

      - 从 **efiboot.img** 挂载点，将 **EFI/redhat/shimx64.efi** 和 **EFI/redhat/grubx64.efi** 文件复制到 TFTP 服务器中：

        ```
        $ cp /mnt/efiboot/EFI/redhat/shimx64.efi .
        ```

        ```
        $ cp /mnt/efiboot/EFI/redhat/grubx64.efi .
        ```

        ```
        $ umount /mnt/efiboot
        ```

        ```
        $ umount /mnt/iso
        ```

      - 将 RHCOS ISO 中包含的 **EFI/redhat/grub.cfg** 文件复制到您的 TFTP 服务器中。

   b. 编辑 **grub.cfg** 文件使其包含类似如下的参数：

      ```
      menuentry 'Install Red Hat Enterprise Linux CoreOS' --class fedora --class gnu-linux --
      class gnu --class os {
       linuxefi rhcos-<version>-live-kernel-<architecture> coreos.inst.install_dev=/dev/sda
      coreos.live.rootfs_url=http://<HTTP_server>/rhcos-<version>-live-rootfs.
      <architecture>.img coreos.inst.ignition_url=http://<HTTP_server>/bootstrap.ign
       initrdefi rhcos-<version>-live-initramfs.<architecture>.img
      }
      ```

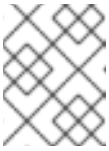
      其中：

      rhcos-<version>-live-kernel-<architecture>
      指定上传到 TFTP 服务器的 **内核** 文件。

      http://<HTTP_server>/rhcos-<version>-live-rootfs.<architecture>.img
      指定上传到 HTTP 服务器的 live rootfs 镜像的位置。

      http://<HTTP_server>/bootstrap.ign
      指定上传到 HTTP 服务器的 bootstrap Ignition 配置文件的位置。

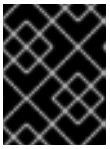
      rhcos-<version>-live-initramfs.<architecture>.img
      指定上传到 TFTP 服务器的 **initramfs** 文件的位置。



注意

有关如何为 UEFI 引导配置 PXE 服务器的更多信息，请参阅红帽知识库文章：如何为 Red Hat Enterprise Linux 的 UEFI 引导配置/设置 PXE 服务器？

7. 继续为集群创建机器。



重要

此刻您必须创建 bootstrap 和 control plane 机器。如果 control plane 机器不可调度（这是默认调度），则在安装集群前至少会创建两台计算机器。

5.1.11.3. 高级 Red Hat Enterprise Linux CoreOS（RHCOS）安装配置

为 OpenShift Container Platform 手动置备 Red Hat Enterprise Linux CoreOS（RHCOS）节点的一个关键优点是能够进行通过默认的 OpenShift Container Platform 安装方法无法进行的配置。本节介绍了您可以使用的一些技术来进行配置，其中包括：

- 将内核参数传递给实时安装程序
- 从 live 系统手动运行 **coreos-installer**
- 将 Ignition 配置嵌入 ISO 中

本节详述了与 Red Hat Enterprise Linux CoreOS（RHCOS）手动安装的高级配置相关的内容，如磁盘分区、网络以及使用 Ignition 配置的不同方式相关。

5.1.11.3.1. 使用高级网络选项进行 PXE 和 ISO 安装

OpenShift Container Platform 节点的网络默认使用 DHCP 来收集所有必要配置设置。要设置静态 IP 地址或配置特殊的设置，如绑定,您可以执行以下操作之一：

- 引导 live 安装程序时会传递特殊的内核参数。
- 使用机器配置将网络文件复制到安装的系统中。
- 使用 live installer shell 提示配置网络，然后将那些设置复制到安装的系统上，以便在安装的系统第一次引导时生效。

要配置 PXE 或 iPXE 安装，请使用以下选项之一：

- 请参阅"高级 RHCOS 安装参考"表。
- 使用机器配置将网络文件复制到安装的系统中。

要配置 ISO 安装，请使用以下步骤。

流程

1. 引导 ISO 安装程序。
2. 在 live 系统 shell 提示下，使用可用的 RHEL 工具（如 **nmcli** 或 **nmtui**）为 Live 系统配置网络。
3. 运行 **coreos-installer** 命令来安装系统，添加 **--copy-network** 选项来复制网络配置。例如：

```
$ coreos-installer install --copy-network \
    --ignition-url=http://host/worker.ign /dev/sda
```

重要

copy-network 选项只复制 **/etc/NetworkManager/system-connections** 下的网络配置。特别是，它不会复制系统主机名。

4. 重启安装的系统。

5.1.11.3.2. 磁盘分区

磁盘分区是在 Red Hat Enterprise Linux CoreOS（RHCOS）安装过程中在 OpenShift Container Platform 集群节点上创建的。特定架构的每个 RHCOS 节点都使用相同的分区布局，除非默认分区配置被覆盖。在 RHCOS 安装过程中，根文件系统的大小会增大，以使用目标设备中剩余的可用空间。

但是，在安装 OpenShift Container Platform 节点时，在两种情况下您可能需要覆盖默认分区：

- 创建单独的分区：对于在空磁盘中的 greenfield 安装，您可能想要在分区中添加单独的存储。这只在生成 **/var** 或者一个 **/var** 独立分区的子目录（如 **/var/lib/etcd** ）时被正式支持，但不支持两者。

重要

Kubernetes 只支持两个文件系统分区。如果您在原始配置中添加多个分区，Kubernetes 无法监控所有这些分区。

- 保留现有分区：对于 brownfield 安装，您要在现有节点上重新安装 OpenShift Container Platform，并希望保留从之前的操作系统中安装的数据分区，对于 **coreos-installer** 来说，引导选项和选项都允许您保留现有数据分区。

5.1.11.3.2.1. 创建一个独立的 /var 分区

通常情况下，OpenShift Container Platform 的磁盘分区应该留给安装程序。然而，在有些情况下您可能需要在文件系统的一部分中创建独立分区。

OpenShift Container Platform 支持添加单个分区来将存储附加到 **/var** 分区或 **/var** 的子目录。例如：

- **/var/lib/containers**：保存镜像相关的内容，随着更多镜像和容器添加到系统中，它所占用的存储会增加。
- **/var/lib/etcd**：保存您可能希望保持独立的数据，比如 etcd 存储的性能优化。
- **/var**：保存您希望独立保留的数据，用于特定目的（如审计）。

单独存储 **/var** 目录的内容可方便地根据需要对区域扩展存储，并可以在以后重新安装 OpenShift Container Platform 时保持该数据地完整。使用这个方法，您不必再次拉取所有容器，在更新系统时也无法复制大量日志文件。

因为 **/var** 在进行一个全新的 Red Hat Enterprise Linux CoreOS（RHCOS）安装前必需存在，所以这个流程会在 OpenShift Container Platform 安装过程的 **openshift-install** 准备阶段插入的机器配置来设置独立的 **/var** 分区。

流程

1. 创建存放 OpenShift Container Platform 安装文件的目录：

```
$ mkdir $HOME/clusterconfig
```

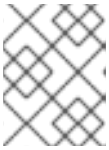
2. 运行 **openshift-install** 在 **manifest** 和 **openshift** 子目录中创建一组文件。在出现提示时回答系统问题：

```
$ openshift-install create manifests --dir $HOME/clusterconfig
? SSH Public Key ...
$ ls $HOME/clusterconfig/openshift/
99_kubeadmin-password-secret.yaml
99_openshift-cluster-api_master-machines-0.yaml
99_openshift-cluster-api_master-machines-1.yaml
99_openshift-cluster-api_master-machines-2.yaml
...
```

3. 创建 **MachineConfig** 对象并将其添加到 **openshift** 目录中的一个文件中。例如，把文件命名为 **98-var-partition.yaml**，将磁盘设备名称改为 **worker** 系统中存储设备的名称，并根据情况设置存储大小。这个示例将 **/var** 目录放在一个单独的分区中：

```
apiVersion: machineconfiguration.openshift.io/v1
kind: MachineConfig
metadata:
  labels:
    machineconfiguration.openshift.io/role: worker
  name: 98-var-partition
spec:
  config:
    ignition:
      version: 3.1.0
    storage:
      disks:
      - device: /dev/<device_name> 1
        partitions:
        - label: var
          startMiB: <partition_start_offset> 2
          sizeMiB: <partition_size> 3
      filesystems:
        - device: /dev/disk/by-partlabel/var
          path: /var
          format: xfs
    systemd:
      units:
        - name: var.mount 4
          enabled: true
          contents: |
            [Unit]
            Before=local-fs.target
            [Mount]
            What=/dev/disk/by-partlabel/var
            Where=/var
            Options=defaults,prjquota 5
            [Install]
            WantedBy=local-fs.target
```

1. 要分区的磁盘的存储设备名称。
2. 当在引导磁盘中添加数据分区时，推荐最少使用 25000MB。root 文件系统会自动重新定义大小使其占据所有可用空间（最多到指定的偏移值）。如果没有指定值，或者指定的值小于推荐的最小值，则生成的 root 文件系统会太小，而在以后进行的 RHCOS 重新安装可能会覆盖数据分区的开始部分。
3. 数据分区的大小（以兆字节为单位）。
4. 挂载单元的名称必须与 **Where=** 指令中指定的目录匹配。例如，对于挂载在 **/var/lib/containers** 上的文件系统，该单元必须命名为 **var-lib-containers.mount**。
5. 对于用于容器存储的文件系统，必须启用 **prjquota** 挂载选项。



注意

在创建独立 **/var** 分区时，如果不同的实例类型没有相同的设备名称，则无法将不同的实例类型用于 worker 节点。

4. 再次运行 **openshift-install**，从 **manifest** 和 **openshift** 子目录中的一组文件创建 Ignition 配置：

```
$ openshift-install create ignition-configs --dir $HOME/clusterconfig
$ ls $HOME/clusterconfig/
auth  bootstrap.ign  master.ign  metadata.json  worker.ign
```

现在，可以使用 Ignition 配置文件作为 ISO 或 PXE 手动安装过程的输入来安装 Red Hat Enterprise Linux CoreOS（RHCOS）系统。

5.1.11.3.2.2. 保留现有分区

对于 ISO 安装，您可以在 **coreos-installer** 命令行中添加可让安装程序维护一个或多个现有分区的选项。对于 PXE 安装，您可以 **APPEND coreos.inst.*** 选项来保留分区。

保存的分区可能是来自现有 OpenShift Container Platform 系统中的分区，其中包括了您希望保留的数据分区。以下是几个提示：

- 如果您保存了现有分区，且这些分区没有为 RHCOS 留下足够空间，则安装将失败但不会损害已保存的分区。
- 通过分区标签或数字识别您要保留的磁盘分区。

对于 ISO 安装

这个示例保留分区标签以**数据**（**data***）开头的任何分区：

```
# coreos-installer install --ignition-url http://10.0.2.2:8080/user.ign \
      --save-partlabel 'data*' /dev/sda
```

以下示例演示了在运行 **coreos-installer** 时要保留磁盘上的第 6 个分区：

```
# coreos-installer install --ignition-url http://10.0.2.2:8080/user.ign \
      --save-partindex 6 /dev/sda
```

这个示例保留了分区 5 及更高分区：

```
# coreos-installer install --ignition-url http://10.0.2.2:8080/user.ign
      --save-partindex 5- /dev/sda
```

在前面已保存分区的示例中，**coreos-installer** 会立即重新创建分区。

对于 PXE 安装

这个 **APPEND** 选项保留分区标签以 'data'('data*')开头的所有分区：

```
coreos.inst.save_partlabel=data*
```

这个 **APPEND** 选项保留分区 5 及其后的分区：

```
coreos.inst.save_partindex=5-
```

这个 **APPEND** 选项保留分区 6:

```
coreos.inst.save_partindex=6
```

5.1.11.3.3. 标识 Ignition 配置

在进行 RHCOS 手动安装时，您可以提供两种 Ignition 配置类型，它们有不同的原因：

- **永久安装 Ignition 配置**：每个手动 RHCOS 安装都需要传递 **openshift-installer** 生成的 Ignition 配置文件之一，如 **bootstrap.ign**、**master.ign** 和 **worker.ign**，才能进行安装。

  重要

  不建议修改这些文件。

  对于 PXE 安装，您可以使用 **coreos.inst.ignition_url=** 选项在 **APPEND** 行上传递 Ignition 配置。对于 ISO 安装，在 ISO 引导至 shell 提示符后，您可以使用 **--ignition-url=** 选项在 **coreos-installer** 命令行上识别 Ignition 配置。在这两种情况下，都只支持 HTTP 和 HTTPS 协议。

- **live 安装 Ignition 配置**：此类型必须手动创建，并应该尽可能避免，因为红帽不支持它。使用此方法，Ignition 配置会传递到 live 安装介质，在引导时立即运行，并在 RHCOS 系统安装到磁盘之前和/或之后执行设置任务。这个方法只用于必须执行一次且之后不能再次应用的任务，如不能使用机器配置进行的高级分区。
  对于 PXE 或 ISO 引导，您可以创建 Ignition 配置，**APPEND ignition.config.url=** 选项，以标识 Ignition 配置的位置。您还需要附加 **ignition.firstboot ignition.platform.id=metal** 或者 **ignition.config.url** 选项。

5.1.11.3.3.1. 在 RHCOS ISO 中嵌入 Ignition 配置

您可以直接嵌入 RHCOS ISO 镜像中的 live 安装 Ignition 配置。引导 ISO 镜像后，内嵌的配置将自动应用。

流程

1. 从以下镜像镜像页面下载 **coreos-installer** 二进制文件：
   https://mirror.openshift.com/pub/openshift-v4/clients/coreos-installer/latest/。
2. 检索 RHCOS ISO 镜像和 Ignition 配置文件，并将其复制到可访问的目录中，如 **/mnt**:
   -

```
# cp rhcos-<version>-live.x86_64.iso bootstrap.ign /mnt/
# chmod 644 /mnt/rhcos-<version>-live.x86_64.iso
```

3. 运行以下命令将 Ignition 配置嵌入 ISO 中：

```
# ./coreos-installer iso ignition embed -i /mnt/bootstrap.ign \
    /mnt/rhcos-<version>-live.x86_64.iso
```

现在，您以使用该 ISO 使用指定的 live 安装 Ignition 配置来安装 RHCOS。

重要

不支持且不推荐使用 **coreos-installer iso ignition embed** 来嵌入由 **openshift-installer** 生成的文件，如 **bootstrap.ign**、**master.ign** 和 **worker.ign**。

4. 要显示嵌入的 Ignition 配置的内容并将其定向到文件中，请运行：

```
# ./coreos-installer iso ignition show /mnt/rhcos-<version>-live.x86_64.iso > mybootstrap.ign
```

```
# diff -s bootstrap.ign mybootstrap.ign
```

输出示例

```
Files bootstrap.ign and mybootstrap.ign are identical
```

5. 要删除 Ignition 配置并将 ISO 返回到其 pristine 状态（因此您可以重复使用它），请运行：

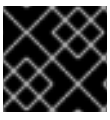
```
# ./coreos-installer iso ignition remove /mnt/rhcos-<version>-live.x86_64.iso
```

现在，您可以将另一个 Ignition 配置嵌入到 ISO 中，或者在其 pristine 状态下使用 ISO。

5.1.11.3.4. 高级 RHCOS 安装参考

本节演示了网络配置和其他高级选项，允许您修改 Red Hat Enterprise Linux CoreOS（RHCOS）手动安装过程。下表描述了您可以与 RHCOS live installer 和 **coreos-installer** 命令一起使用的内核参数和命令行选项。

RHCOS 启动提示下的路由和绑定选项
如果从 ISO 镜像安装 RHCOS，您可以在引导该镜像时手动添加内核参数以配置节点的网络。如果没有使用网络参数，则安装默认为使用 DHCP。



重要

添加网络参数时，还必须添加 **rd.neednet=1** 内核参数。

下表描述了如何为实时 ISO 安装使用 **ip=**、**nameserver=** 和 **bond=** 内核参数。

注意

在添加内核参数时顺序非常重要：**ip=**，**nameserver=**，然后 **bond=**。

ISO 的路由和绑定选项

下表提供了配置 Red Hat Enterprise Linux CoreOS（RHCOS）节点网络的示例。这些是在系统引导过程中传递给 **dracut** 工具的网络选项。有关 **dracut** 支持的网络选项的详情，请参考 **dracut.cmdline** 手册页。

描述	例子
要配置一个 IP 地址，可以使用 DHCP(**ip=dhcp**)或者设置单独的静态 IP 地址(**ip=<host_ip>**)。然后在每个节点上指定 DNS 服务器 IP 地址(**nameserver=<dns_ip>**)。这个示例设置： • 节点的 IP 地址为 **10.10.10.2** • 网关地址为 **10.10.10.254** • 子网掩码为 **255.255.255.0** • 主机名为 **core0.example.com** • DNS 服务器地址为 **4.4.4.41**	ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp1s0:none nameserver=4.4.4.41
通过指定多个 **ip=** 条目来指定多个网络接口。	ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp1s0:none ip=10.10.10.3::10.10.10.254:255.255.255.0:core0.example.com:enp2s0:none
可选：您可以通过设置一个 **rd.route=** 值来配置到额外网络的路由。 如果额外网络网关与主要网络网关不同，则默认网关必须是主要网络网关。	配置默认网关： ip=::10.10.10.254:::: 为额外网络配置路由： rd.route=20.20.20.0/24:20.20.20.254:enp2s0
在单一接口中禁用 DHCP，比如当有两个或者多个网络接口时，且只有一个接口被使用。	ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp1s0:none ip=::::core0.example.com:enp2s0:none
您可以将系统中 DHCP 和静态 IP 配置与多个网络接口结合在一起。	ip=enp1s0:dhcp ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp2s0:none

描述	例子
可选： 您可以使用 **vlan=** 参数在单独的接口上配置 VLAN。	在网络接口中配置 VLAN 并使用静态 IP 地址： ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp2s0.100:none vlan=enp2s0.100:enp2s0 在网络接口中配置 VLAN 并使用 DHCP： ip=enp2s0.100:dhcp vlan=enp2s0.100:enp2s0
您可以为每个服务器添加一个 **nameserver=** 条目来提供多个 DNS 服务器。	nameserver=1.1.1.1 nameserver=8.8.8.8
可选：使用 **bond=** 选项支持将多个网络接口绑定到一个接口。在这两个示例中： - 配置绑定接口的语法为：**bond=name[:network_interfaces][:options]** - *name* 是绑定设备名称（**bond0**），*network_interfaces* 代表用逗号分开的物理（以太网）接口(**em1,em2**)的列表，*options* 是用逗号分开的绑定选项列表。输入 **modinfo bonding** 查看可用选项。 - 当使用 **bond=** 创建绑定接口时，您必须指定如何分配 IP 地址以及绑定接口的其他信息。	要将绑定的接口配置为使用 DHCP，请将绑定的 IP 地址设置为 **dhcp**。例如： bond=bond0:em1,em2:mode=active-backup ip=bond0:dhcp 要将绑定接口配置为使用静态 IP 地址，请输入您需要的特定 IP 地址以及相关信息。例如： bond=bond0:em1,em2:mode=active-backup ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:bond0:none
可选： 您可以使用 **vlan=** 参数在绑定接口上配置 VLAN。	使用 VLAN 配置绑定接口并使用 DHCP： ip=bond0.100:dhcp bond=bond0:em1,em2:mode=active-backup vlan=bond0.100:bond0 使用 VLAN 配置绑定接口，并使用静态 IP 地址： ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:bond0.100:none bond=bond0:em1,em2:mode=active-backup vlan=bond0.100:bond0

描述	例子
可选：通过使用 **team=** 参数，网络合作可用作绑定的替代选择。在此例中： • 配置组接口的语法为：**team=name[:network_interfaces]** *name* 是组设备名称(**team0**)，**network** *_interfaces* 代表以逗号分隔的物理（以太网）接口（**em1**、**em2**）列表。  **注意** 当 RHCOS 切换到即将推出的 RHEL 版本时，团队计划会被弃用。如需更多信息，请参阅红帽知识库文章。	配置网络团队： team=team0:em1,em2 ip=team0:dhcp

coreos.inst 引导选项用于 ISO 或 PXE 安装

虽然您可以将大多数标准安装引导参数传递给 live 安装程序，但也有一些特定于 RHCOS live 安装程序的参数。

- 对于 ISO，可以通过中断 RHCOS 安装程序来添加这些选项。
- 对于 PXE 或 iPXE，这些选项必须在启动 PXE 内核前添加到 **APPEND** 行中。您无法中断实时 PXE 安装。

下表显示了用于 ISO 和 PXE 安装的 RHCOS live installer 引导选项。

表 5.11. coreos.inst 引导选项

参数	描述
coreos.inst.install_dev	必需。要安装的系统中的块设备。虽然可以使用 **sda** 这样的相对路径，但建议使用完整路径，如 **/dev/sda**。
coreos.inst.ignition_url	可选：嵌入到已安装系统中的 Ignition 配置的 URL如果没有指定 URL，则不会嵌入 Ignition 配置。
coreos.inst.save_partlabel	可选：在安装过程中要保留的分区压缩标签。允许使用 glob 风格的通配符。指定分区不需要存在。
coreos.inst.save_partindex	可选：在安装过程中完成要保留的分区的分离索引。可以使用 **m-n** 指定范围，**m** 或 **n** 可以被省略。指定分区不需要存在。
coreos.inst.insecure	可选：将 **coreos.inst.image_url** 指定的 OS 镜像提交取消签名。

参数	描述
coreos.inst.image_url	可选：下载并安装指定的 RHCOS 镜像。 ● 这个参数不应该在生产环境中使用，而是只用于调试目的。 ● 虽然在 RHCOS 的安装版本与 live 介质的版本不匹配时可以使用这个参数，但建议使用与您要安装版本匹配的介质。 ● 如果您使用的是 **coreos.inst.image_url**。还必须使用 **coreos.inst.insecure**。这是因为，裸机介质没有为 OpenShift Container Platform 进行 GPG 签名。 ● 只支持 HTTP 和 HTTPS 协议。
coreos.inst.skip_reboot	可选：安装后该系统不会重启。安装完成后，您会收到提示，提示您检查在安装过程中发生的情况。这个参数不应该在生产环境中使用，而是只用于调试目的。
coreos.inst.platform_id	可选：安装 RHCOS 镜像的平台的 Ignition 平台 ID。默认为 **metal**。这个选项决定是否从云供应商（如 VMware）请求 Ignition 配置。例如：**coreos.inst.platform_id=vmware**。
ignition.config.url	可选：用于实时启动的 Ignition 配置的 URL。例如，它可以用来定制调用 **coreos-installer** 的方式，或者用来在安装前或安装后运行代码。这与 **coreos.inst.ignition_url**（这是已安装系统的 Ignition 配置）不同。

ISO 安装的 coreos-installer 选项

您还可以直接从命令行调用 **coreos-installer** 命令来安装 RHCOS。上表中的内核参数提供了在引导时自动调用 **coreos-installer** 的快捷方式，但您可以在 shell 提示符运行时将类似的参数直接传递给 **coreos-installer**。

下表显示了您可以在实时安装过程中从 shell 提示符传递给 **coreos-installer** 命令的选项和子命令。

表 5.12. CoreOS-installer 命令行选项、参数和子命令

命令行选项	
选项	**描述**
-u, **--image-url <url>**	手动指定镜像 URL。
-f, **--image-file <path>**	手动指定本地镜像文件。
-i, --ignition-file <path>	从文件中嵌入 Ignition 配置。

-I, --ignition-url <URL>	从 URL 嵌入 Ignition 配置。
--ignition-hash <digest>	Ignition config 的 **type-value** 的文摘值。
-p, --platform <name>	覆盖 Ignition 平台 ID。
--append-karg <arg>…	附加默认内核参数。
--delete-karg <arg>…	删除默认内核参数。
-n, --copy-network	从安装环境中复制网络配置。  **重要** **copy-network** 选项只复制 **/etc/NetworkManager/system-connections** 下的网络配置。特别是，它不会复制系统主机名。
--network-dir <path>	使用 **-n**。默认为 **/etc/NetworkManager/system-connections/**。
--save-partlabel <lx>..	使用这个标签 glob 保存分区。
--save-partindex <id>…	使用这个数值或者范围保存分区。
--offline	强制离线安装。
--insecure	跳过签名验证。
--insecure-ignition	允许没有 HTTPS 或 hash 的 Ignition URL。
--architecture <name>	目标 CPU 架构。默认为 **x86_64**。
--preserve-on-error	出现错误时不清除分区表。
-h，--help	打印帮助信息。
命令行参数	
参数	**描述**
<device>	目的设备。
CoreOS-installer 嵌入的 Ignition 命令	
命令	**描述**

$ coreos-installer iso ignition embed <options> --ignition-file <file_path> <ISO_image>	在 ISO 镜像中嵌入 Ignition 配置。
coreos-installer iso ignition show <options> <ISO_image>	显示来自 ISO 镜像的内嵌 Ignition 配置。
coreos-installer iso ignition remove <options> <ISO_image>	从 ISO 镜像中删除嵌入的 Ignition 配置。

coreos-installer ISO Ignition 选项

选项	**描述**
-f, --force	覆盖现有的 Ignition 配置。
-i, --ignition-file <path>	要使用的 Ignition 配置。默认为 **stdin**。
-o, --output <path>	将 ISO 写入到一个新输出文件。
-h, --help	打印帮助信息。

coreos-installer PXE Ignition 命令

命令	**描述**
请注意，不是所有子命令都接受这些选项。	
coreos-installer pxe ignition wrap <options>	在镜像中嵌套 Ignition 配置。
coreos-installer pxe ignition unwrap <options> <image_name>	显示在镜像中嵌套的 Ignition 配置。
coreos-installer pxe ignition unwrap <options> <initrd_name>	在 **initrd** 镜像中显示嵌套的 Ignition 配置。

coreos-installer PXE Ignition 选项

选项	**描述**
-i, --ignition-file <path>	要使用的 Ignition 配置。默认为 **stdin**。
-o, --output <path>	将 ISO 写入到一个新输出文件。
-h, --help	打印帮助信息。

5.1.12. 创建集群

要创建 OpenShift Container Platform 集群，请等待您通过安装程序生成的 Ignition 配置文件所置备的机器上完成 bootstrap 过程。

先决条件

- 为集群创建所需的基础架构。
- 已获得安装程序并为集群生成了 Ignition 配置文件。
- 已使用 Ignition 配置文件为集群创建 RHCOS 机器。
- 您的机器可直接访问互联网，或者可以使用 HTTP 或 HTTPS 代理。

流程

1. 监控 bootstrap 过程：

```
$ ./openshift-install --dir <installation_directory> wait-for bootstrap-complete \ 1
    --log-level=info 2
```

1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

2 要查看不同的安装详情，请指定 **warn**、**debug** 或 **error**，而不要指定 **info**。

输出示例

```
INFO Waiting up to 30m0s for the Kubernetes API at https://api.test.example.com:6443...
INFO API v1.19.0 up
INFO Waiting up to 30m0s for bootstrapping to complete...
INFO It is now safe to remove the bootstrap resources
```

Kubernetes API 服务器提示已在 control plane 机器上完成 bootstrap 时，命令运行成功。

2. bootstrap 过程完成后，请从负载均衡器中删除 bootstrap 机器。

重要

此时您必须从负载均衡器中删除 bootstrap 机器。您还可以删除或重新格式化机器本身。

5.1.13. 使用 CLI 登录到集群

您可以通过导出集群 **kubeconfig** 文件，以默认系统用户身份登录集群。**kubeconfig** 文件包含关于集群的信息，供 CLI 用于将客户端连接到正确集群和 API 服务器。该文件特只适用于一个特定的集群，在 OpenShift Container Platform 安装过程中创建。

先决条件

- 已部署了 OpenShift Container Platform 集群。
- 已安装 **oc** CLI。

流程

1. 导出 **kubeadmin** 凭证：

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

2. 使用导出的配置，验证能否成功运行 **oc** 命令：

```
$ oc whoami
```

输出示例

```
system:admin
```

5.1.14. 批准机器的证书签名请求

将机器添加到集群时，会为您添加的每台机器生成两个待处理证书签名请求（CSR）。您必须确认这些 CSR 已获得批准，或根据需要自行批准。客户端请求必须首先被批准，然后是服务器请求。

先决条件

- 您已将机器添加到集群中。

流程

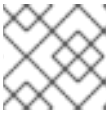
1. 确认集群可以识别这些机器：

```
$ oc get nodes
```

输出示例

```
NAME     STATUS   ROLES   AGE  VERSION
master-0 Ready    master  63m  v1.19.0
master-1 Ready    master  63m  v1.19.0
master-2 Ready    master  64m  v1.19.0
```

输出将列出您创建的所有机器。



注意

在一些 CSR 被批准前，以上输出可能不包括计算节点（也称为 worker 节点）。

2. 检查待处理的 CSR，并确保可以看到添加到集群中的每台机器都有 **Pending** 或 **Approved** 状态的客户端请求：

```
$ oc get csr
```

输出示例

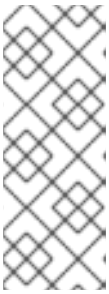
```
NAME        AGE    REQUESTOR                                                   CONDITION
csr-8b2br   15m    system:serviceaccount:openshift-machine-config-operator:node-bootstrapper   Pending
csr-8vnps   15m    system:serviceaccount:openshift-machine-config-operator:node-bootstrapper   Pending
...
```

在本例中，两台机器加入了集群。您可能会在列表中看到更多已批准的 CSR。

3. 如果 CSR 没有获得批准，请在所添加机器的所有待处理 CSR 都处于 **Pending** 状态后，为您的集群机器批准这些 CSR：

注意

由于 CSR 会自动轮转，因此请在将机器添加到集群后一小时内批准您的 CSR。如果没有在一小时内批准，证书将会轮转，每个节点将会存在多个证书。您必须批准所有这些证书。批准客户端 CSR 后， Kubelet 为服务证书创建辅助 CSR，这需要手动批准。然后，如果 Kubelet 请求具有相同参数的新证书，则 **machine-approver** 会自动批准后续服务证书续订请求。



注意

对于在未启用机器 API 的平台中运行的集群，如裸机和其他用户置备的基础架构，必须采用一种方法自动批准 kubelet 提供证书请求（CSR）。如果没有批准请求，则 **oc exec**、**oc rsh** 和 **oc logs** 命令将无法成功，因为 API 服务器连接到 kubelet 时需要服务证书。与 Kubelet 端点联系的任何操作都需要此证书批准。这个方法必须监视新的 CSR，确认 CSR 由 **system:node** 或 **system:admin** 组中的 **node-bootstrapper** 服务帐户提交，并确认节点的身份。

- 若要单独批准，请对每个有效的 CSR 运行以下命令：

```
$ oc adm certificate approve <csr_name> 1
```

1 **<csr_name>** 是当前 CSR 列表中 CSR 的名称。

- 要批准所有待处理的 CSR，请运行以下命令：

```
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}{{"\n"}}{{end}}{{end}}' | xargs --no-run-if-empty oc adm certificate approve
```

注意

在有些 CSR 被批准前，一些 Operator 可能无法使用。

4. 现在，您的客户端请求已被批准，您必须查看添加到集群中的每台机器的服务器请求：

```
$ oc get csr
```

输出示例

```
NAME        AGE    REQUESTOR                                                   CONDITION
```

```
csr-bfd72   5m26s   system:node:ip-10-0-50-126.us-east-2.compute.internal
Pending
csr-c57lv   5m26s   system:node:ip-10-0-95-157.us-east-2.compute.internal
Pending
...
```

5. 如果剩余的 CSR 没有被批准，且处于 **Pending** 状态，请批准集群机器的 CSR：

- 若要单独批准，请对每个有效的 CSR 运行以下命令：

```
$ oc adm certificate approve <csr_name> 1
```

1 **<csr_name>** 是当前 CSR 列表中 CSR 的名称。

- 要批准所有待处理的 CSR，请运行以下命令：

```
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}{{"\n"}}
{{end}}{{end}}' | xargs oc adm certificate approve
```

6. 批准所有客户端和服务器 CSR 后，器将处于 **Ready** 状态。运行以下命令验证：

```
$ oc get nodes
```

输出示例

```
NAME      STATUS    ROLES   AGE  VERSION
master-0  Ready     master  73m  v1.20.0
master-1  Ready     master  73m  v1.20.0
master-2  Ready     master  74m  v1.20.0
worker-0  Ready     worker  11m  v1.20.0
worker-1  Ready     worker  11m  v1.20.0
```

注意

批准服务器 CSR 后可能需要几分钟时间让机器转换为 **Ready** 状态。

其他信息

- 如需有关 CSR 的更多信息，请参阅证书签名请求。

5.1.15. 初始 Operator 配置

在 control plane 初始化后，您必须立即配置一些 Operator 以便它们都可用。

先决条件

- 您的 control plane 已初始化。

流程

1. 观察集群组件上线：

```
$ watch -n5 oc get clusteroperators
```

输出示例

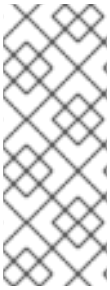
```
NAME                                       VERSION AVAILABLE   PROGRESSING   DEGRADED
SINCE
authentication                             4.6.0   True        False         False      3h56m
cloud-credential                           4.6.0   True        False         False      29h
cluster-autoscaler                         4.6.0   True        False         False      29h
config-operator                            4.6.0   True        False         False      6h39m
console                                    4.6.0   True        False         False      3h59m
csi-snapshot-controller                    4.6.0   True        False         False      4h12m
dns                                        4.6.0   True        False         False      4h15m
etcd                                       4.6.0   True        False         False      29h
image-registry                             4.6.0   True        False         False      3h59m
ingress                                    4.6.0   True        False         False      4h30m
insights                                   4.6.0   True        False         False      29h
kube-apiserver                             4.6.0   True        False         False      29h
kube-controller-manager                    4.6.0   True        False         False      29h
kube-scheduler                             4.6.0   True        False         False      29h
kube-storage-version-migrator              4.6.0   True        False         False      4h2m
machine-api                                4.6.0   True        False         False      29h
machine-approver                           4.6.0   True        False         False      6h34m
machine-config                             4.6.0   True        False         False      3h56m
marketplace                                4.6.0   True        False         False      4h2m
monitoring                                 4.6.0   True        False         False      6h31m
network                                    4.6.0   True        False         False      29h
node-tuning                                4.6.0   True        False         False      4h30m
openshift-apiserver                        4.6.0   True        False         False      3h56m
openshift-controller-manager               4.6.0   True        False         False      4h36m
openshift-samples                          4.6.0   True        False         False      4h30m
operator-lifecycle-manager                 4.6.0   True        False         False      29h
operator-lifecycle-manager-catalog         4.6.0   True        False         False      29h
operator-lifecycle-manager-packageserver   4.6.0   True        False         False      3h59m
service-ca                                 4.6.0   True        False         False      29h
storage                                    4.6.0   True        False         False      4h30m
```

2. 配置不可用的 Operator。

5.1.15.1. 安装过程中删除的镜像 registry

在不提供可共享对象存储的平台上，OpenShift Image Registry Operator bootstraps 本身的状态是 **Removed**。这允许 **openshift-installer** 在这些平台类型上完成安装。

将 **ManagementState** Image Registry Operator 配置从 **Removed** 改为 **Managed**。



注意

Prometheus 控制台提供了一个 **ImageRegistryRemoved** 警报，例如：

"Image Registry has been removed.**ImageStreamTags**, **BuildConfigs** and **DeploymentConfigs** which reference **ImageStreamTags** may not work as expected.Please configure storage and update the config to **Managed** state by editing configs.imageregistry.operator.openshift.io."

5.1.15.2. 镜像 registry 存储配置

对于不提供默认存储的平台，Image Registry Operator 最初将不可用。安装后，您必须配置 registry 使用的存储，这样 Registry Operator 才可用。

示配置生产集群所需的持久性卷的说明。如果适用，显示有关将空目录配置为存储位置的说明，该位置只可用于非生产集群。

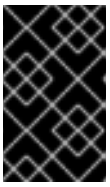
另外还提供了在升级过程中使用 **Recreate** rollout 策略来允许镜像 registry 使用块存储类型的说明。

5.1.15.2.1. 为裸机和其他手动安装配置 registry 存储

作为集群管理员，在安装后需要配置 registry 来使用存储。

先决条件

- 具有 Cluster Administrator 权限
- 使用手动置备的 Red Hat Enterprise Linux CoreOS（RHCOS）节点（如裸机）的集群。
- 为集群置备的持久性存储，如 Red Hat OpenShift Container Storage。



重要

如果您只有一个副本，OpenShift Container Platform 支持对镜像 registry 存储的 **ReadWriteOnce** 访问。要部署支持高可用性的、带有两个或多个副本的镜像 registry，需要 **ReadWriteMany** 访问设置。

- 必须具有 100Gi 容量。

流程

1. 为了配置 registry 使用存储，需要修改 **configs.imageregistry/cluster** 资源中的 **spec.storage.pvc** 。

注意

使用共享存储时，请查看您的安全设置以防止被外部访问。

2. 验证您没有 registry pod:

```
$ oc get pod -n openshift-image-registry
```

注意

如果存储类型为 **emptyDIR**，则副本数不能超过 **1**。

3. 检查 registry 配置：

```
$ oc edit configs.imageregistry.operator.openshift.io
```

输出示例

-

```
storage:
  pvc:
    claim:
```

将 **claim** 字段留空以允许自动创建一个 **image-registry-storage** PVC。

4. 检查 **clusteroperator** 的状态：

```
$ oc get clusteroperator image-registry
```

5. 确保您的 registry 设置为 manage，以启用镜像的构建和推送。

- 运行：

```
$ oc edit configs.imageregistry/cluster
```

然后将行改

```
managementState: Removed
```

为

```
managementState: Managed
```

5.1.15.2.2. 在非生产集群中配置镜像 registry 存储

您必须为 Image Registry Operator 配置存储。对于非生产集群，您可以将镜像 registry 设置为空目录。如果您这样做，重启 registry 后会丢失所有镜像。

流程

- 将镜像 registry 存储设置为空目录：

```
$ oc patch configs.imageregistry.operator.openshift.io cluster --type merge --patch '{"spec":{"storage":{"emptyDir":{}}}}'
```

警告

仅可为非生产集群配置这个选项。

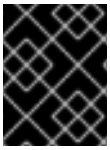
如果在 Image Registry Operator 初始化其组件前运行此命令，**oc patch** 命令会失败并显示以下错误：

```
Error from server (NotFound): configs.imageregistry.operator.openshift.io "cluster" not found
```

等待几分钟，然后再次运行该命令。

5.1.15.2.3. 配置块 registry 存储

要允许镜像 registry 在作为集群管理员升级过程中使用块存储类型，您可以使用 **Recreate** rollout 策略。



重要

支持块存储卷，但不建议将其与生产环境中的镜像 registry 一起使用。在块存储上配置 registry 的安装不具有高可用性，因为 registry 无法拥有多个副本。

流程

1. 要将镜像 registry 存储设置为块存储类型，对 registry 进行补丁，使其使用 **Recreate** rollout 策略，并只使用一个（**1**）副本运行：

```
$ oc patch config.imageregistry.operator.openshift.io/cluster --type=merge -p '{"spec":
{"rolloutStrategy":"Recreate","replicas":1}}'
```

2. 为块存储设备置备 PV，并为该卷创建 PVC。请求的块卷使用 ReadWriteOnce（RWO）访问模式。

3. 编辑 registry 配置，使其引用正确的 PVC。

5.1.16. 在用户置备的基础架构上完成安装

完成 Operator 配置后，可以在您提供的基础架构上完成集群安装。

先决条件

- 您的 control plane 已初始化。
- 已完成初始 Operator 配置。

流程

1. 使用以下命令确认所有集群组件都已在线：

```
$ watch -n5 oc get clusteroperators
```

输出示例

```
NAME                       VERSION AVAILABLE   PROGRESSING   DEGRADED
SINCE
authentication               4.6.0   True        False         False      3h56m
cloud-credential             4.6.0   True        False         False      29h
cluster-autoscaler           4.6.0   True        False         False      29h
config-operator              4.6.0   True        False         False      6h39m
console                    4.6.0   True        False         False      3h59m
csi-snapshot-controller          4.6.0   True        False         False      4h12m
dns                        4.6.0   True        False         False      4h15m
etcd                       4.6.0   True        False         False      29h
image-registry               4.6.0   True        False         False      3h59m
ingress                    4.6.0   True        False         False      4h30m
insights                   4.6.0   True        False         False      29h
kube-apiserver               4.6.0   True        False         False      29h
```

```
kube-controller-manager                    4.6.0   True        False         False      29h
kube-scheduler                             4.6.0   True        False         False      29h
kube-storage-version-migrator              4.6.0   True        False         False      4h2m
machine-api                                4.6.0   True        False         False      29h
machine-approver                           4.6.0   True        False         False      6h34m
machine-config                             4.6.0   True        False         False      3h56m
marketplace                                4.6.0   True        False         False      4h2m
monitoring                                 4.6.0   True        False         False      6h31m
network                                    4.6.0   True        False         False      29h
node-tuning                                4.6.0   True        False         False      4h30m
openshift-apiserver                        4.6.0   True        False         False      3h56m
openshift-controller-manager               4.6.0   True        False         False      4h36m
openshift-samples                          4.6.0   True        False         False      4h30m
operator-lifecycle-manager                 4.6.0   True        False         False      29h
operator-lifecycle-manager-catalog         4.6.0   True        False         False      29h
operator-lifecycle-manager-packageserver   4.6.0   True        False         False      3h59m
service-ca                                 4.6.0   True        False         False      29h
storage                                    4.6.0   True        False         False      4h30m
```

或者，通过以下命令，如果所有集群都可用您会接到通知。它还检索并显示凭证：

```
$ ./openshift-install --dir <installation_directory> wait-for install-complete 1
```

1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

输出示例

```
INFO Waiting up to 30m0s for the cluster to initialize...
```

Cluster Version Operator 完成从 Kubernetes API 服务器部署 OpenShift Container Platform 集群时，命令运行成功。

重要

- 安装程序生成的 Ignition 配置文件包含在 24 小时后过期的证书，然后在过期时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外情况是，您需要手动批准待处理的 **node-bootstrapper** 证书签名请求（CSR）来恢复 kubelet 证书。如需更多信息，请参阅*从过期的 control plane 证书中恢复*的文档。
- 建议您在生成 12 小时后使用 Ignition 配置文件，因为集群安装后 24 小时证书从 16 小时轮转至 22 小时。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中运行证书更新时避免安装失败。

2. 确认 Kubernetes API 服务器正在与 pod 通信。

   a. 要查看所有 pod 的列表，请使用以下命令：

   ```
   $ oc get pods --all-namespaces
   ```

   输出示例

   ```
   NAMESPACE                  NAME                                        READY   STATUS
   ```

```
RESTARTS   AGE
openshift-apiserver-operator      openshift-apiserver-operator-85cb746d55-zqhs8   1/1
Running   1          9m
openshift-apiserver               apiserver-67b9g                                 1/1     Running     0
3m
openshift-apiserver               apiserver-ljcmx                                 1/1     Running     0
1m
openshift-apiserver               apiserver-z25h4                                 1/1     Running     0
2m
openshift-authentication-operator authentication-operator-69d5d8bf84-vh2n8        1/1
Running     0          5m
...
```

b. 使用以下命令，查看上一命令的输出中所列 pod 的日志：

```
$ oc logs <pod_name> -n <namespace> 1
```

1 指定 pod 名称和命名空间，如上一命令的输出中所示。

如果 pod 日志显示，Kubernetes API 服务器可以与集群机器通信。

5.1.17. OpenShift Container Platform 的 Telemetry 访问

在 OpenShift Container Platform 4.6 中，默认运行的 Telemetry 服务提供有关集群健康状况和成功更新的指标，需要访问互联网。如果您的集群连接到互联网，Telemetry 会自动运行，而且集群会注册到 OpenShift Cluster Manager。

确认 OpenShift Cluster Manager 清单正确后，可以由 Telemetry 自动维护，也可以使用 OpenShift Cluster Manager 手动维护，使用订阅监控来跟踪帐户或多集群级别的 OpenShift Container Platform 订阅。

其他资源

- 有关 Telemetry 服务的更多信息，请参阅关于远程健康监控。

5.1.18. 后续步骤

- 自定义集群。
- 如果需要，您可以选择不使用远程健康报告。
- 设置 registry 并配置 registry 存储。

5.2. 使用网络自定义在裸机上安装集群

在 OpenShift Container Platform 版本 4.6 中，您可以使用自定义的网络配置选项在裸机环境中安装集群。通过自定义网络配置，您的集群可以与环境中现有的 IP 地址分配共存，并与现有的 MTU 和 VXLAN 配置集成。

大部分网络配置参数必须在安装过程中设置，只有 **kubeProxy** 配置参数可以在运行的集群中修改。

5.2.1. 先决条件

- 查看有关 OpenShift Container Platform 安装和更新流程的详细信息。
- 如果使用防火墙，您必须将其配置为访问 Red Hat Insights。

5.2.2. OpenShift Container Platform 的互联网访问

在 OpenShift Container Platform 4.6 中，您需要访问互联网来安装集群。

您必须具有以下互联网访问权限：

- 访问 OpenShift Cluster Manager 以下载安装程序并执行订阅管理。如果集群可以访问互联网，并且没有禁用 Telemetry，该服务会自动授权您的集群。
- 访问 Quay.io，以获取安装集群所需的软件包。
- 获取执行集群更新所需的软件包。

重要

如果您的集群无法直接访问互联网，则可以在置备的某些类基础架构上执行受限网络安装。在此过程中，您要下载所需的内容，并使用它在镜像 registry（mirror registry） 中填充安装集群并生成安装程序所需的软件包。对于某些安装类型，集群要安装到的环境不需要访问互联网。在更新集群之前，要更新 registry 镜像系统中的内容。

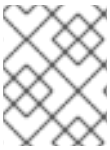
5.2.3. 具有用户置备基础架构的集群的机器要求

对于含有用户置备的基础架构的集群，您必须部署所有所需的机器。

5.2.3.1. 所需的机器

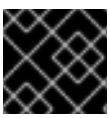
最小的 OpenShift Container Platform 集群需要下列主机：

- 一个临时 bootstrap 机器
- 三台 control plane 或 master 机器
- 至少两台计算机器，也称为 worker 机器。如果您正在运行三节点集群，则支持运行零个计算机器。不支持运行一台计算机器。



注意

集群要求 bootstrap 机器在三台 control plane 机器上部署 OpenShift Container Platform 集群。您可在安装集群后删除 bootstrap 机器。



重要

要保持集群的高可用性，请将独立的物理主机用于这些集群机器。

bootstrap 和 control plane 机器必须使用 Red Hat Enterprise Linux CoreOS (RHCOS) 作为操作系统。但是，计算机器可以在 Red Hat Enterprise Linux CoreOS(RHCOS)或 Red Hat Enterprise Linux(RHEL)7.9 之间进行选择。

请注意，RHCOS 基于 Red Hat Enterprise Linux（RHEL） 8，并继承其所有硬件认证和要求。请查看Red Hat Enterprise Linux 技术功能及限制 。

5.2.3.2. 网络连接要求

所有 Red Hat Enterprise Linux CoreOS (RHCOS) 机器在启动过程中需要 **initramfs** 中的网络从 Machine Config Server 获取 Ignition 配置文件。在初次启动过程中，需要一个 DHCP 服务器或设置了静态 IP 地址来建立网络连接，以下载它们的 Ignition 配置文件。另外，集群中的每个 OpenShift Container Platform 节点都必须有权访问网络时间协议（NTP）服务器。如果 DHCP 服务器提供 NTP 服务器信息，Red Hat Enterprise Linux CoreOS（RHCOS）机器上的 chrony 时间服务会读取信息，并可与 NTP 服务器同步时钟。

5.2.3.3. 最低资源要求

每台集群机器都必须满足以下最低要求：

表 5.13. 最低资源要求

机器	操作系统	CPU [1]	RAM	存储	IOPS [2]
bootstrap	RHCOS	4	16 GB	100 GB	300
Control plane	RHCOS	4	16 GB	100 GB	300
Compute	RHCOS 或 RHEL 7.9	2	8 GB	100 GB	300

1. 当未启用并发多线程(SMT)或超线程时，一个 CPU 相当于一个物理内核。启用后，使用以下公式来计算对应的比率：（每个内核的线程数）¹ 插槽 = CPU。
2. OpenShift Container Platform 和 Kubernetes 对磁盘性能非常敏感，建议使用更快的存储速度，特别是 control plane 节点上需要 10 ms p99 fsync 持续时间的 etcd。请注意，在许多云平台上，存储大小和 IOPS 可一起扩展，因此您可能需要过度分配存储卷来获取足够的性能。

5.2.3.4. 证书签名请求管理

在使用您置备的基础架构时，集群只能有限地访问自动机器管理，因此您必须提供一种在安装后批准集群证书签名请求 (CSR) 的机制。**kube-controller-manager** 只能批准 kubelet 客户端 CSR。 **machine-approver** 无法保证使用 kubelet 凭证请求的提供证书的有效性，因为它不能确认是正确的机器发出了该请求。您必须决定并实施一种方法，以验证 kubelet 提供证书请求的有效性并进行批准。

5.2.4. 创建用户置备的基础架构

在部署采用用户置备的基础架构的 OpenShift Container Platform 集群前，您必须创建底层基础架构。

先决条件

- 在为集群创建支持基础架构之前，请参阅OpenShift Container Platform 4.x Tested Integrations页。

流程

1. 在每个节点上配置 DHCP 或设置静态 IP 地址。
2. 提供所需的负载均衡器。

3. 配置机器的端口。
4. 配置 DNS。
5. 确保网络可以正常工作。

5.2.4.1. 用户置备的基础架构对网络的要求

所有 Red Hat Enterprise Linux CoreOS（RHCOS）机器在启动过程中需要 **initramfs** 中的网络从机器配置服务器获取 Ignition 配置。

在初次启动过程中，需要一个 DHCP 服务器或集群中的每个机器都设置了静态 IP 地址来建立网络连接，以下载它们的 Ignition 配置文件。

建议您使用 DHCP 服务器为集群进行长期机器管理。确保 DHCP 服务器已配置为向集群机器提供持久 IP 地址和主机名。

Kubernetes API 服务器必须能够解析集群机器的节点名称。如果 API 服务器和 worker 节点位于不同的区域中，您可以配置默认 DNS 搜索区域，以便 API 服务器能够解析节点名称。另一种支持的方法是始终在节点对象和所有 DNS 请求中使用完全限定域名来指代主机。

您必须配置机器间的网络连接，以便集群组件进行通信。每台机器都必须能够解析集群中所有其他机器的主机名。

表 5.14. 所有机器到所有机器

协议	端口	描述
ICMP	N/A	网络可访问性测试
TCP	**1936**	指标
	9000-**9999**	主机级别的服务，包括端口 **9100**-**9101** 上的节点导出器和端口 **9099** 上的 Cluster Version Operator。
	10250-**10259**	Kubernetes 保留的默认端口
	10256	openshift-sdn
UDP	**4789**	VXLAN 和 Geneve
	6081	VXLAN 和 Geneve
	9000-**9999**	主机级别的服务，包括端口 **9100**-**9101** 上的节点导出器。
TCP/UDP	**30000**-**32767**	Kubernetes 节点端口

表 5.15. 要通过控制平面的所有机器

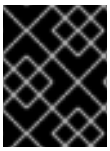
协议	端口	描述
TCP	**6443**	Kubernetes API

表 5.16. control plane 机器到 control plane 机器

协议	端口	描述
TCP	**2379-2380**	etcd 服务器和对等端口

网络拓扑要求

您为集群置备的基础架构必须满足下列网络拓扑要求。



重要

OpenShift Container Platform 要求所有节点都能访问互联网，以便为平台容器提取镜像并向红帽提供遥测数据。

负载均衡器

在安装 OpenShift Container Platform 前，您必须置备两个满足以下要求的负载均衡器：

1. **API 负载均衡器**：提供一个通用端点，供用户（包括人和机器）与平台交互和配置。配置以下条件：
   - 只适用于第 4 层负载均衡。这可被称为 Raw TCP、SSL Passthrough 或者 SSL 桥接模式。如果使用 SSL Bridge 模式，必须为 API 路由启用 Server Name Indication（SNI）。
   - 无状态负载平衡算法。这些选项根据负载均衡器的实现而有所不同。

重要

不要为 API 负载均衡器配置会话持久性。

在负载均衡器的前端和后台配置以下端口：

表 5.17. API 负载均衡器

端口	后端机器（池成员）	内部	外部	描述
6443	Bootstrap 和 control plane.bootstrap 机器初始化集群 control plane 后，您要从负载均衡器中删除 bootstrap 机器。您必须为 API 服务器健康检查探测配置 **/readyz** 端点。	X	X	Kubernetes API 服务器
22623	Bootstrap 和 control plane.bootstrap 机器初始化集群 control plane 后，您要从负载均衡器中删除 bootstrap 机器。	X		机器配置服务器

注意

负载均衡器必须配置为，从 API 服务器关闭 **/readyz** 端点到从池中删除 API 服务器实例时最多需要 30 秒。在 **/readyz** 返回错误或处于健康状态后的时间范围内，端点必须被删除或添加。每 5 秒或 10 秒探测一次，有两个成功请求处于健康状态，三个成为不健康的请求经过测试。

2. **应用程序入口负载均衡器**:提供来自集群外部的应用程序流量流量的 Ingress 点。配置以下条件：

   - 只适用于第 4 层负载均衡。这可被称为 Raw TCP、SSL Passthrough 或者 SSL 桥接模式。如果使用 SSL Bridge 模式，您必须为 Ingress 路由启用Server Name Indication（SNI）。
   - 建议根据可用选项以及平台上托管的应用程序类型，使用基于连接的或者基于会话的持久性。

   在负载均衡器的前端和后台配置以下端口：

表 5.18. 应用程序入口负载均衡器

端口	后端机器（池成员）	内部	外部	描述
443	默认运行入口路由器 Pod、计算或 worker 的机器。	X	X	HTTPS 流量
80	默认运行入口路由器 Pod、计算或 worker 的机器。	X	X	HTTP 流量

提示

如果负载均衡器可以看到客户端的真实 IP 地址，启用基于 IP 的会话持久性可提高使用端到端 TLS 加密的应用程序的性能。

注意

OpenShift Container Platform 集群需要正确配置入口路由器。control plane 初始化后，您必须配置入口路由器。

NTP 配置

OpenShift Container Platform 集群默认配置为使用公共网络时间协议（NTP）服务器。如果要使用本地企业 NTP 服务器，或者集群部署在断开连接的网络中，您可以将集群配置为使用特定的时间服务器。如需更多信息，请参阅*配置 chrony 时间服务*的文档。

如果 DHCP 服务器提供 NTP 服务器信息，Red Hat Enterprise Linux CoreOS（RHCOS）机器上的 chrony 时间服务会读取信息，并可与 NTP 服务器同步时钟。

其他资源

- 配置 chrony 时间服务

5.2.4.2. 用户置备 DNS 要求

DNS 用于名称解析和反向名称解析。DNS A/AAAA 或 CNAME 记录用于名称解析，PTR 记录用于反向解析名称。反向记录很重要，因为 Red Hat Enterprise Linux CoreOS（RHCOS）使用反向记录为所有节点

设置主机名。另外，反向记录用于生成 OpenShift Container Platform 需要操作的证书签名请求（CSR）。

采用用户置备的基础架构的 OpenShift Container Platform 集群需要以下 DNS 记录。在每一记录中，**<cluster_name>** 是集群名称，**<base_domain>** 则是您在 **install-config.yaml** 文件中指定的集群基域。完整的 DNS 记录采用如下格式: **<component>.<cluster_name>.<base_domain>.**。

表 5.19. 所需的 DNS 记录

组件	记录	描述
Kubernetes API	**api.<cluster_name>. <base_domain>.**	添加 DNS A/AAAA 或 CNAME 记录，以及 DNS PTR 记录，以识别 control plane 机器的负载均衡器。这些记录必须由集群外的客户端以及集群中的所有节点解析。
	api-int.<cluster_name>. <base_domain>.	添加 DNS A/AAAA 或 CNAME 记录，以及 DNS PTR 记录，以识别 control plane 机器的负载均衡器。这些记录必须可以从集群中的所有节点解析。  **重要** API 服务器必须能够根据在 Kubernetes 中记录的主机名解析 worker 节点。如果 API 服务器无法解析节点名称，则代理的 API 调用会失败，且您无法从 pod 检索日志。
Routes	***.apps.<cluster_name>. <base_domain>.**	添加通配符 DNS A/AAAA 或 CNAME 记录，指向以运行入口路由器 Pod 的机器（默认为 worker 节点）为目标的负载均衡器。这些记录必须由集群外的客户端以及集群中的所有节点解析。
bootstrap	**bootstrap.<cluster_name>. <base_domain>.**	添加 DNS A/AAAA 或 CNAME 记录，以及 DNS PTR 记录来识别 bootstrap 机器。这些记录必须由集群中的节点解析。
Master 主机	**<master><n>. <cluster_name>. <base_domain>.**	DNS A/AAAA 或 CNAME 记录，以识别 control plane 节点（也称为 master 节点）的每台机器。这些记录必须由集群中的节点解析。
Worker 主机	**<worker><n>. <cluster_name>. <base_domain>.**	添加 DNS A/AAAA 或 CNAME 记录，以识别 worker 节点的每台机器。这些记录必须由集群中的节点解析。

提示

您可以使用 **nslookup <hostname>** 命令来验证名称解析。您可以使用 **dig -x <ip_address>** 命令来验证 PTR 记录的反向名称解析。

下面的 BIND 区文件的例子展示了关于名字解析的 A 记录的例子。这个示例的目的是显示所需的记录。这个示例不是为选择一个名称解析服务提供建议。

例 5.3. DNS 区数据库示例

```
$TTL 1W
@ IN SOA ns1.example.com. root (
   2019070700 ; serial
   3H  ; refresh (3 hours)
   30M  ; retry (30 minutes)
   2W  ; expiry (2 weeks)
   1W )  ; minimum (1 week)
 IN NS ns1.example.com.
 IN MX 10 smtp.example.com.
;
;
ns1 IN A 192.168.1.5
smtp IN A 192.168.1.5
;
helper IN A 192.168.1.5
helper.ocp4 IN A 192.168.1.5
;
; The api identifies the IP of your load balancer.
api.ocp4  IN A 192.168.1.5
api-int.ocp4  IN A 192.168.1.5
;
; The wildcard also identifies the load balancer.
*.apps.ocp4  IN A 192.168.1.5
;
; Create an entry for the bootstrap host.
bootstrap.ocp4 IN A 192.168.1.96
;
; Create entries for the master hosts.
master0.ocp4  IN A 192.168.1.97
master1.ocp4  IN A 192.168.1.98
master2.ocp4  IN A 192.168.1.99
;
; Create entries for the worker hosts.
worker0.ocp4  IN A 192.168.1.11
worker1.ocp4  IN A 192.168.1.7
;
;EOF
```

下面的 BIND 区文件示例显示了反向名字解析的 PTR 记录示例。

例 5.4. 反向记录的 DNS 区数据库示例

```
$TTL 1W
@ IN SOA ns1.example.com. root (
   2019070700 ; serial
   3H  ; refresh (3 hours)
   30M  ; retry (30 minutes)
   2W  ; expiry (2 weeks)
   1W )  ; minimum (1 week)
 IN NS ns1.example.com.
;
; The syntax is "last octet" and the host must have an FQDN
; with a trailing dot.
97 IN PTR master0.ocp4.example.com.
```

```
98 IN PTR master1.ocp4.example.com.
99 IN PTR master2.ocp4.example.com.
;
96 IN PTR bootstrap.ocp4.example.com.
;
5 IN PTR api.ocp4.example.com.
5 IN PTR api-int.ocp4.example.com.
;
11 IN PTR worker0.ocp4.example.com.
7 IN PTR worker1.ocp4.example.com.
;
;EOF
```

5.2.5. 生成 SSH 私钥并将其添加到代理中

如果要在集群上执行安装调试或灾难恢复，则必须为 **ssh-agent** 和安装程序提供 SSH 密钥。您可以使用此密钥访问公共集群中的 bootstrap 机器来排除安装问题。

注意

在生产环境中，您需要进行灾难恢复和调试。

您可以使用此密钥以 **core** 用户身份通过 SSH 连接到 master 节点。在部署集群时，此密钥会添加到 **core** 用户的 **~/.ssh/authorized_keys** 列表中。

注意

您必须使用一个本地密钥，而不要使用在特定平台上配置的密钥，如 AWS 密钥对。

流程

1. 如果还没有为计算机上免密码身份验证而配置的 SSH 密钥，请创建一个。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ ssh-keygen -t ed25519 -N '' \
  -f <path>/<file_name> 1
```

1 指定新 SSH 密钥的路径和文件名，如 **~/.ssh/id_rsa**。如果您已有密钥对，请确保您的公钥位于 **~/.ssh** 目录中。

运行此命令会在指定的位置生成不需要密码的 SSH 密钥。

注意

如果您计划在 **x86_64** 架构中安装使用 FIPS 验证的/Modules in Process 加密库的 OpenShift Container Platform 集群，不要创建使用 **ed25519** 算法的密钥。反之，创建一个使用 **rsa** 或 **ecdsa** 算法的密钥。

2. 作为后台任务启动 **ssh-agent** 进程：

```
$ eval "$(ssh-agent -s)"
```

-

输出示例

```
Agent pid 31874
```

注意

如果您的集群采用 FIPS 模式，则只使用 FIPS 兼容算法来生成 SSH 密钥。密钥必须是 RSA 或 ECDSA。

3. 将 SSH 私钥添加到 **ssh-agent**：

```
$ ssh-add <path>/<file_name> 1
```

输出示例

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

1 指定 SSH 私钥的路径和文件名，如 **~/.ssh/id_rsa**

后续步骤

- 在安装 OpenShift Container Platform 时，为安装程序提供 SSH 公钥。

5.2.6. 获取安装程序

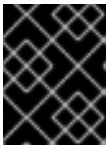
在安装 OpenShift Container Platform 之前，将安装文件下载到本地计算机上。

先决条件

- 运行 Linux 或 macOS 的计算机，本地磁盘空间为 500 MB

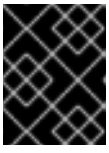
流程

1. 访问 OpenShift Cluster Manager 站点的 Infrastructure Provider 页面。如果您有红帽帐号，请使用自己的凭证登录。如果没有，请创建一个帐户。
2. 选择您的基础架构供应商。
3. 进入适用于您的安装类型的页面，下载您的操作系统的安装程序，并将文件放在要保存安装配置文件的目录中。。



重要

安装程序会在用来安装集群的计算机上创建若干文件。在完成集群安装后，您必须保留安装程序和安装程序所创建的文件。这两个文件都需要删除集群。



重要

删除安装程序创建的文件不会删除您的集群，即使集群在安装过程中失败也是如此。要删除集群，为特定云供应商完成 OpenShift Container Platform 卸载流程。

4. 提取安装程序。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ tar xvf openshift-install-linux.tar.gz
```

5. 从 Red Hat OpenShift Cluster Manager 下载安装 pull secret 。通过此 pull secret，您可以进行所含授权机构提供的服务的身份验证，这些服务包括为 OpenShift Container Platform 组件提供容器镜像的 Quay.io。

5.2.7. 通过下载二进制文件安装 OpenShift CLI

您需要安装 CLI（**oc**） 来使用命令行界面与 OpenShift Container Platform 进行交互。您可在 Linux 、Windows 或 macOS 上安装 **oc**。

重要

如果安装了旧版本的 **oc**，则无法使用 OpenShift Container Platform 4.6 中的所有命令。下载并安装新版本的 **oc**。

5.2.7.1. 在 Linux 上安装 OpenShift CLI

您可以按照以下流程在 Linux 上安装 OpenShift CLI（**oc**）二进制文件。

流程

1. 进入到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 在 **Version** 下拉菜单中选择相应的版本。
3. 单击 **OpenShift v4.6 Linux 客户端** 条目旁边的 **Download Now**，再保存文件。
4. 解包存档：

```
$ tar xvzf <file>
```

5. 把 **oc** 二进制代码放到 **PATH** 中的目录中。
执行以下命令可以查看当前的 **PATH** 设置：

```
$ echo $PATH
```

安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
$ oc <command>
```

5.2.7.2. 在 Windows 上安装 OpenShift CLI

您可以按照以下流程在 Windows 上安装 OpenShift CLI（**oc**）二进制代码。

流程

1. 进入到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 在 **Version** 下拉菜单中选择相应的版本。

3. 单击 **OpenShift v4.6 Windows 客户端**条目旁边的 **Download Now**，再保存文件。
4. 使用 ZIP 程序解压存档。
5. 把 **oc** 二进制代码放到 **PATH** 中的目录中。
   要查看您的 **PATH**，请打开命令提示窗口并执行以下命令：

```
C:\> path
```

安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
C:\> oc <command>
```

5.2.7.3. 在 macOS 上安装 OpenShift CLI

您可以按照以下流程在 macOS 上安装 OpenShift CLI（**oc**）二进制代码。

流程

1. 进入到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 在 **Version** 下拉菜单中选择相应的版本。
3. 单击 **OpenShift v4.6 MacOSX 客户端**条目旁边的 **Download Now**，再保存文件。
4. 解包和解压存档。
5. 将 **oc** 二进制文件移到 PATH 的目录中。
   要查看您的 **PATH**，打开一个终端窗口并执行以下命令：

```
$ echo $PATH
```

安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
$ oc <command>
```

5.2.8. 手动创建安装配置文件

对于使用用户置备的基础架构的 OpenShift Container Platform 安装，您必须手动生成安装配置文件。

先决条件

- 获取 OpenShift Container Platform 安装程序和集群的访问令牌。

流程

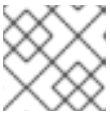
1. 创建用来存储您所需的安装资产的安装目录：

```
$ mkdir <installation_directory>
```

重要

您必须创建目录。一些安装信息，如 bootstrap X.509 证书，有较短的过期间隔，因此不要重复使用安装目录。如果要重复使用另一个集群安装中的个别文件，可以将其复制到您的目录中。但是，一些安装数据的文件名可能会在发行版本之间有所改变。从 OpenShift Container Platform 老版本中复制安装文件时要格外小心。

2. 自定义以下 **install-config.yaml** 文件模板，并将它保存到 **<installation_directory>** 中。



注意

此配置文件必须命名为 **install-config.yaml**。

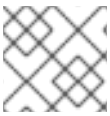
3. 备份 **install-config.yaml** 文件，以便用于安装多个集群。

重要

install-config.yaml 文件会在安装过程的下一步骤中消耗掉。现在必须备份它。

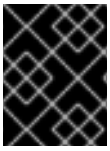
5.2.8.1. 安装配置参数

在部署 OpenShift Container Platform 集群前，您可以提供参数值，以描述托管集群的云平台的帐户并选择性地自定义集群平台。在创建 **install-config.yaml** 安装配置文件时，您可以通过命令行来提供所需的参数的值。如果要自定义集群，可以修改 **install-config.yaml** 文件来提供关于平台的更多信息。



注意

安装之后，您无法修改 **install-config.yaml** 文件中的这些参数。



重要

openshift-install 命令不验证参数的字段名称。如果指定了不正确的名称，则不会创建相关的文件或对象，且不会报告错误。确保所有指定的参数的字段名称都正确。

5.2.8.1.1. 所需的配置参数

下表描述了所需的安装配置参数：

表 5.20. 所需的参数

参数	描述	值
apiVersion	**install-config.yaml** 内容的 API 版本。当前版本是 **v1**。安装程序还可能支持旧的 API 版本。	字符串

参数	描述	值
baseDomain	云供应商的基域。此基础域用于创建到 OpenShift Container Platform 集群组件的路由。集群的完整 DNS 名称是 **baseDomain** 和 **metadata.name** 参数值的组合，其格式为 **<metadata.name>.<baseDomain>**。	完全限定域名或子域名，如 **example.com**。
metadata	Kubernetes 资源 **ObjectMeta**，其中只消耗 **name** 参数。	对象
metadata.name	集群的名称。集群的 DNS 记录是 **{{.metadata.name}}.{{.baseDomain}}** 的子域。	小写字母,连字符(-)和句点(.)的字符串，如 **dev**。
platform	执行安装的具体平台配置：**aws**、**baremetal**、**azure**、**openstack**、**ovirt**、**vsphere**。有关 **platform.<platform>** 参数的额外信息，请参考下表来了解您的具体平台。	对象

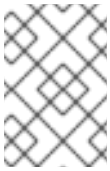
pullSecret	从 Red Hat OpenShift Cluster Manager 获取 pull secret，验证从 Quay.io 等服务中下载 OpenShift Container Platform 组件的容器镜像。	{
		"auths":{
		"cloud.openshift.com":{
		"auth":"b3Blb=",
		"email":"you@example.com"
		},
		"quay.io":{
		"auth":"b3Blb=",
		"email":"you@example.com"
		}
		}
		}

5.2.8.1.2. 网络配置参数

您可以根据现有网络基础架构的要求自定义安装配置。例如，您可以扩展集群网络的 IP 地址块，或者提供不同于默认值的不同 IP 地址块。

只支持 IPv4 地址。

表 5.21. 网络参数

参数	描述	值
networking	集群网络的配置。	对象  **注意** 您不能在安装后修改 **networking** 对象指定的参数。
networking.networkType	要安装的集群网络供应商 Container Network Interface（CNI）插件。	**OpenShiftSDN** 或 **OVNKubernetes**。默认值为 **OpenShiftSDN**。
networking.clusterNetwork	pod 的 IP 地址块。 默认值为 **10.128.0.0/14**，主机前缀为 **/23**。 如果您指定多个 IP 地址块，则块不得互相重叠。	一个对象数组。例如： networking: clusterNetwork: - cidr: 10.128.0.0/14 hostPrefix: 23
networking.clusterNetwork.cidr	使用 **networking.clusterNetwork** 时需要此项。IP 地址块。 一个 IPv4 网络。	使用 CIDR 形式的 IP 地址块。IPv4 块的前缀长度介于 **0** 到 **32** 之间。
networking.clusterNetwork.hostPrefix	分配给每个单独节点的子网前缀长度。例如，如果 **hostPrefix** 设为 **23**，则每个节点从所给的 **cidr** 中分配一个 **/23** 子网。**hostPrefix** 值 **23** 提供 510（2^(32 - 23)- 2）个 pod IP 地址。	子网前缀。 默认值为 **23**。
networking.serviceNetwork	服务的 IP 地址块。默认值为 **172.30.0.0/16**。 OpenShift SDN 和 OVN-Kubernetes 网络供应商只支持服务网络的一个 IP 地址块。	CIDR 格式具有 IP 地址块的数组。例如： networking: serviceNetwork: - 172.30.0.0/16
networking.machineNetwork	机器的 IP 地址块。 如果您指定多个 IP 地址块，则块不得互相重叠。	一个对象数组。例如： networking: machineNetwork: - cidr: 10.0.0.0/16

参数	描述	值
networking.machineNetwork.cidr	使用 **networking.machineNetwork** 时需要。IP 地址块。libvirt 以外的所有平台的默认值为 **10.0.0.0/16**。对于 libvirt，默认值为 **192.168.126.0/24**。	CIDR 表示法中的 IP 网络块。 例如： **10.0.0.0/16**。  **注意** 将 **networking.machineNetwork** 设置为与首选 NIC 所在的 CIDR 匹配。

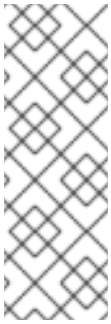
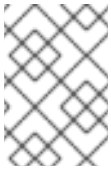
5.2.8.1.3. 可选配置参数

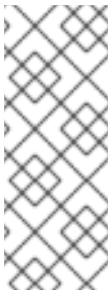
下表描述了可选安装配置参数：

表 5.22. 可选参数

参数	描述	值
additionalTrustBundle	添加到节点可信证书存储中的 PEM 编码 X.509 证书捆绑包。配置了代理时，也可以使用这个信任捆绑包。	字符串
compute	组成计算节点的机器的配置。	machine-pool 对象的数组。详情请查看以下"Machine-pool"表。
compute.architecture	决定池中机器的指令集合架构。目前不支持异构集群，因此所有池都必须指定相同的架构。有效值为 **amd64** （默认值）。	字符串
compute.hyperthreading	是否在计算机器上启用或禁用并发多线程或**超线程**。默认情况下，启用并发多线程以提高机器内核的性能。  **重要** 如果禁用并发多线程，请确保在容量规划时考虑到机器性能可能会显著降低的问题。	**Enabled** 或 **Disabled**
compute.name	使用 **compute** 时需要此值。机器池的名称。	**worker**

参数	描述	值
compute.platform	使用 **compute** 时需要此值。使用此参数指定托管 worker 机器的云供应商。此参数值必须与 **controlPlane.platform** 参数值匹配。	**aws**、**azure**、**gcp**、**openstack**、**ovirt**、**vsphere** 或 **{}**
compute.replicas	要置备的计算机器数量，也称为 worker 机器。	大于或等于 **2** 的正整数。默认值为 **3**。
controlPlane	组成 control plane 的机器的配置。	**MachinePool** 对象的数组。详情请查看以下"Machine-pool"表。
controlPlane.architecture	决定池中机器的指令集合架构。目前不支持异构集群，因此所有池都必须指定相同的架构。有效值为 **amd64** （默认值）。	字符串
controlPlane.hyperthreading	是否在 control plane 机器上启用或禁用并发多线程或**超线程**。默认情况下，启用并发多线程以提高机器内核的性能。  **重要** 如果禁用并发多线程，请确保在容量规划时考虑到机器性能可能会显著降低的问题。	**Enabled** 或 **Disabled**
controlPlane.name	使用 **controlPlane** 时需要。机器池的名称。	**master**
controlPlane.platform	使用 **controlPlane** 时需要。使用此参数指定托管 control plane 机器的云供应商。此参数值必须与 **compute.platform** 参数值匹配。	**aws**、**azure**、**gcp**、**openstack**、**ovirt**、**vsphere** 或 **{}**
controlPlane.replicas	要置备的 control plane 机器数量。	唯一支持的值是 **3**，它是默认值。

参数	描述	值
credentialsMode	Cloud Credential Operator（CCO）模式。如果没有指定任何模式，CCO 会动态地尝试决定提供的凭证的功能，在支持多个模式的平台上使用 mint 模式。  **注意** 不是所有 CCO 模式都支持所有云供应商。如需有关 CCO 模式的更多信息，请参阅*Red Hat Operator 参考指南*内容中的 *Cloud Credential Operator* 条目。	**Mint**、**Passthrough**、**Manual** 或空字符串(**""**)。
fips	启用或禁用 FIPS 模式。默认为 **false**（禁用）。如果启用了 FIPS 模式，运行 OpenShift Container Platform 的 Red Hat Enterprise Linux CoreOS (RHCOS) 机器会绕过默认的 Kubernetes 加密套件，并使用由 RHCOS 提供的加密模块。  **重要** 只有在 **x86_64** 架构中的 OpenShift Container Platform 部署支持 FIPS 验证的/Modules in Process 加密库。  **注意** 如果使用 Azure File 存储，则无法启用 FIPS 模式。	**false** 或 **true**
imageContentSources	release-image 内容的源和仓库。	对象数组。包括一个 **source** 以及可选的 **mirrors**，如下表所示。

参数	描述	值
imageContentSources.source	使用 **imageContentSources** 时需要。指定用户在镜像拉取规格中引用的仓库。	字符串
imageContentSources.mirrors	指定可能还包含同一镜像的一个或多个仓库。	字符串数组
publish	如何发布或公开集群的面向用户的端点，如 Kubernetes API、OpenShift 路由。	**Internal** 或 **External**。默认值为 **External**。 在非云平台上不支持将此字段设置为 **Internal**。  **重要** 如果将字段的值设为 **Internal**，集群将无法运行。如需更多信息，请参阅 BZ#1953035。
sshKey	用于验证集群机器访问的 SSH 密钥或密钥。  **注意** 对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 **ssh-agent** 进程使用的 SSH 密钥。	一个或多个密钥。例如： sshKey: <key1> <key2> <key3>

5.2.8.2. 裸机 install-config.yaml 文件示例

您可以自定义 **install-config.yaml** 文件，以指定有关 OpenShift Container Platform 集群平台的更多信息，或修改所需参数的值。

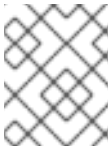
```
apiVersion: v1
baseDomain: example.com (1)
compute: (2)
- hyperthreading: Enabled (3)
  name: worker
  replicas: 0 (4)
controlPlane: (5)
  hyperthreading: Enabled (6)
  name: master
```

```
  replicas: 3 7
metadata:
  name: test 8
networking:
  clusterNetwork:
  - cidr: 10.128.0.0/14 9
    hostPrefix: 23 10
  networkType: OpenShiftSDN
  serviceNetwork: 11
  - 172.30.0.0/16
platform:
  none: {} 12
fips: false 13
pullSecret: '{"auths": ...}' 14
sshKey: 'ssh-ed25519 AAAA...' 15
```

1 集群的基域。所有 DNS 记录都必须是这个基域的子域，并包含集群名称。

2 5 **controlPlane** 部分是一个单映射，但 **compute** 部分是一系列映射。为满足不同数据结构的要求，**compute** 部分的第一行必须以连字符 **-** 开头，**controlPlane** 部分的第一行则不可以连字符开头。只使用一个 control plane 池。

3 6 是否要启用或禁用并发多线程（SMT）或**超线程**。默认情况下，启用 SMT 可提高机器内核的性能。您可以通过将参数值设为 **Disabled** 来禁用。如果禁用 SMT，则必须在所有集群机器中禁用它，其中包括 control plane 和计算机器。



注意

默认启用并发多线程（SMT）。如果在 BIOS 设置中没有启用 SMT，**hyperthreading** 参数不会起作用。

重要

如果您禁用 **hyperthreading**（无论是在 BIOS 中还是在 **install-config.yaml** 中），请确保您对可能会造成的机器性能显著降低的情况有所考虑。

4 **replicas** 参数的值必须设置为 **0**。此参数控制集群为您创建和管理的 worker 数量，使用用户置备的基础架构时集群不会执行这些功能。在完成 OpenShift Container Platform 安装前，您必须手动为集群部署 worker 机器。

7 您添加到集群的 control plane 机器数量。由于集群将这个值用作集群中 etcd 端点的数量，因此该值必须与您部署的 control plane 机器数量匹配。

8 您在 DNS 记录中指定的集群名称。

9 从中分配 pod IP 地址的 IP 地址块。此块不得与现有的物理网络重叠。这些 IP 地址用于 pod 网络。如果您需要从外部网络访问 pod，请配置负载均衡器和路由器来管理流量。

注意

类 E CIDR 范围保留给以后使用。要使用 Class E CIDR 范围，您必须确保您的网络环境接受 Class E CIDR 范围内的 IP 地址。

10 分配给每个单独节点的子网前缀长度。例如，如果 **hostPrefix** 设为 **23**，则每个节点从所给的 **cidr** 中分配一个 **/23** 子网，这样就能有 510 (2^(32 - 23) - 2) 个 Pod IP 地址。如果您需要从外部网络访

11 用于服务 IP 地址的 IP 地址池。您只能输入一个 IP 地址池。此块不得与现有的物理网络重叠。如果您需要从外部网络访问服务，请配置负载均衡器和路由器来管理流量。

12 您必须将平台设置为 **none**。您不能为您的平台提供额外的平台配置变量。

13 是否启用或禁用 FIPS 模式。默认情况下不启用 FIPS 模式。如果启用了 FIPS 模式，运行 OpenShift Container Platform 的 Red Hat Enterprise Linux CoreOS (RHCOS) 机器会绕过默认的 Kubernetes 加密套件，并使用由 RHCOS 提供的加密模块。

重要

只有在 **x86_64** 架构中的 OpenShift Container Platform 部署支持 FIPS 验证的/Modules in Process 加密库。

14 Red Hat OpenShift Cluster Manager 中的 pull secret 。通过此 pull secret，您可以进行所含授权机构提供的服务的身份验证，这些服务包括为 OpenShift Container Platform 组件提供容器镜像的 Quay.io。

15 Red Hat Enterprise Linux CoreOS (RHCOS) 中 **core** 用户的默认 SSH 密钥的公钥部分。

注意

对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 **ssh-agent** 进程使用的 SSH 密钥。

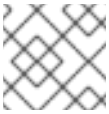
5.2.9. 网络配置阶段

当在安装前指定集群配置时，在安装过程中的几个阶段可以修改网络配置：

阶段 1

输入 **openshift-install create install-config** 命令后。在 **install-config.yaml** 文件中，您可以自定义以下与网络相关的字段：

- **networking.networkType**
- **networking.clusterNetwork**
- **networking.serviceNetwork**
- **networking.machineNetwork**
  有关这些字段的更多信息，请参阅"安装配置参数"。



注意

将 **networking.machineNetwork** 设置为与首选 NIC 所在的 CIDR 匹配。

阶段 2

输入 **openshift-install create manifests** 命令后。如果必须指定高级网络配置，在这个阶段中，只能使用您要修改的字段来定义自定义的 Cluster Network Operator 清单。

在 2 阶段，您无法覆盖 **install-config.yaml** 文件中的 1 阶段中指定的值。但是，您可以在第 2 阶段进一步自定义集群网络供应商。

5.2.10. 指定高级网络配置

您可以通过为集群网络供应商指定额外的配置，使用高级配置自定义将集群整合到现有网络环境中。您只能在安装集群前指定高级网络配置。

重要

不支持修改安装程序创建的 OpenShift Container Platform 清单文件。支持应用您创建的清单文件，如以下流程所示。

先决条件

- 创建 **install-config.yaml** 文件并完成对其所做的任何修改。
- 为集群生成 Ignition 配置文件。

流程

1. 进入包含安装程序的目录并创建清单：

```
$ ./openshift-install create manifests --dir <installation_directory>
```

其中：

<installation_directory>
指定包含集群的 **install-config.yaml** 文件的目录名称。

2. 在 **<installation_directory>/manifests/** 目录下，为高级网络配置创建一个名为 **cluster-network-03-config.yml** 的 stub 清单文件：

```
$ cat <<EOF > <installation_directory>/manifests/cluster-network-03-config.yml
apiVersion: operator.openshift.io/v1
kind: Network
metadata:
  name: cluster
spec:
EOF
```

其中：

<installation_directory>
指定包含集群的 **manifests/** 目录的目录名称。

3. 在编辑器中打开 **cluster-network-03-config.yml** 文件，并为集群指定高级网络配置，如下例所示：

为 OpenShift SDN 网络供应商指定不同的 VXLAN 端口

```
apiVersion: operator.openshift.io/v1
kind: Network
metadata:
```

```
  name: cluster
spec:
  defaultNetwork:
    openshiftSDNConfig:
      vxlanPort: 4800
```

4. 保存 **cluster-network-03-config.yml** 文件，再退出文本编辑器。

5. 可选：备份 **manifests/cluster-network-03-config.yml** 文件。创建集群时，安装程序会删除 **manifests/** 目录。

5.2.11. Cluster Network Operator 配置

集群网络的配置作为 Cluster Network Operator (CNO) 配置的一部分被指定，并存储在名为 **cluster**的自定义资源（CR）对象中。CR 指定 **operator.openshift.io** API 组中的 **Network** API 的字段。

CNO 配置会在集群安装过程中从 **Network.config.openshift.io** API 组中的 **Network** API 继承以下字段，这些字段无法更改：

clusterNetwork

从中分配 pod IP 地址的 IP 地址池。

serviceNetwork

服务的 IP 地址池。

defaultNetwork.type

集群网络供应商，如 OpenShift SDN 或 OVN-Kubernetes。

您可以通过在名为 **cluster** 的 CNO 对象中设置 **defaultNetwork** 对象的字段来为集群指定集群网络供应商配置。

5.2.11.1. Cluster Network Operator 配置对象

Cluster Network Operator（CNO）的字段在下表中描述：

表 5.23. Cluster Network Operator 配置对象

字段	类型	描述
metadata.name	**字符串**	CNO 对象的名称。这个名称始终是 **cluster**。

spec.clusterNetwork	**数组**	用于指定从哪些 IP 地址块分配 Pod IP 地址以及分配给集群中每个节点的子网前缀长度的列表。例如：
		`spec:`
		`  clusterNetwork:`
		`  - cidr: 10.128.0.0/19`
		`  hostPrefix: 23`
		`  - cidr: 10.128.32.0/19`
		`  hostPrefix: 23`
		此值是只读的，并在 **install-config.yaml** 文件中指定。

字段	类型	描述
spec.serviceNetwork	**数组**	服务的 IP 地址块。OpenShift SDN 和 OVN-Kubernetes Container Network Interface（CNI）网络供应商只支持服务网络具有单个 IP 地址块。例如： spec: serviceNetwork: - 172.30.0.0/14 此值是只读的，并在 **install-config.yaml** 文件中指定。
spec.defaultNetwork	**对象**	为集群网络配置 Container Network Interface（CNI）集群网络供应商。
spec.kubeProxyConfig	**对象**	此对象的字段指定 kube-proxy 配置。如果您使用 OVN-Kubernetes 集群网络供应商，则 kube-proxy 的配置不会起作用。

defaultNetwork 对象配置
defaultNetwork 对象的值在下表中定义：

表 5.24. defaultNetwork 对象

字段	类型	描述
type	**字符串**	**OpenShiftSDN** 或 **OVNKubernetes**。在安装过程中选择了集群网络供应商。集群安装后无法更改这个值。 **注意** OpenShift Container Platform 默认使用 OpenShift SDN Container Network Interface（CNI）集群网络供应商。
openshiftSDNConfig	**对象**	此对象仅对 OpenShift SDN 集群网络供应商有效。
ovnKubernetesConfig	**对象**	此对象仅对 OVN-Kubernetes 集群网络供应商有效。

配置 OpenShift SDN CNI 集群网络供应商
下表描述了 OpenShift SDN Container Network Interface（CNI）集群网络供应商的配置字段。

表 5.25. openshiftSDNConfig 对象

字段	类型	描述
mode	**字符串**	配置 OpenShift SDN 的网络隔离模式。默认值为 **NetworkPolicy**。 **Multitenant** 和 **Subnet** 的值可以向后兼容 OpenShift Container Platform 3.x，但不推荐这样做。集群安装后无法更改这个值。
mtu	**整数**	VXLAN 覆盖网络的最大传输单元 (MTU) 。这根据主网络接口的 MTU 自动探测。您通常不需要覆盖检测到的 MTU。 如果自动探测的值不是您期望的，请确认节点上主网络接口中的 MTU 是正确的。您不能使用这个选项更改节点上主网络接口的 MTU 值。 如果您的集群中的不同节点需要不同的 MTU 值，则必须将此值设置为比集群中的最低 MTU 值小 **50**。例如，如果集群中的某些节点的 MTU 为 **9001**，而某些节点的 MTU 为 **1500**，则必须将此值设置为 **1450**。 集群安装后无法更改这个值。
vxlanPort	**整数**	用于所有 VXLAN 数据包的端口。默认值为 **4789**。集群安装后无法更改这个值。 如果您在虚拟环境中运行，并且现有节点是另一个 VXLAN 网络的一部分，那么可能需要更改此值。例如，当在 VMware NSX-T 上运行 OpenShift SDN 覆盖时，您必须为 VXLAN 选择一个备用端口，因为两个 SDN 都使用相同的默认 VXLAN 端口号。 在 Amazon Web Services (AWS) 上，您可以在端口 **9000** 和端口 **9999** 之间为 VXLAN 选择一个备用端口。

OpenShift SDN 配置示例

```
defaultNetwork:
  type: OpenShiftSDN
  openshiftSDNConfig:
    mode: NetworkPolicy
    mtu: 1450
    vxlanPort: 4789
```

配置 OVN-Kubernetes CNI 集群网络供应商

下表描述了 OVN-Kubernetes CNI 集群网络供应商的配置字段。

表 5.26. ovnKubernetesConfig 对象

字段	类型	描述

字段	类型	描述
mtu	**整数**	Geneve（Generic Network Virtualization Encapsulation）覆盖网络的最大传输单元（MTU）。这根据主网络接口的 MTU 自动探测。您通常不需要覆盖检测到的 MTU。 如果自动探测的值不是您期望的，请确认节点上主网络接口中的 MTU 是正确的。您不能使用这个选项更改节点上主网络接口的 MTU 值。 如果您的集群中的不同节点需要不同的 MTU 值，则必须将此值设置为比集群中的最低 MTU 值小 **100**。例如，如果集群中的某些节点的 MTU 为 **9001**，而某些节点的 MTU 为 **1500**，则必须将此值设置为 **1400**。 集群安装后无法更改这个值。
genevePort	**整数**	用于所有 Geneve 数据包的端口。默认值为 **6081**。集群安装后无法更改这个值。

OVN-Kubernetes 配置示例

```
defaultNetwork:
  type: OVNKubernetes
  ovnKubernetesConfig:
    mtu: 1400
    genevePort: 6081
```

kubeProxyConfig 对象配置

kubeProxyConfig 对象的值在下表中定义：

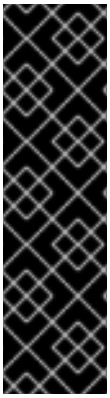
表 5.27. kubeProxyConfig 对象

字段	类型	描述
iptablesSyncPeriod	**字符串**	**iptables** 规则的刷新周期。默认值为 **30s**。有效的后缀包括 **s**、**m** 和 **h**，具体参见 Go **time** 软件包文档。  **注意** 由于 OpenShift Container Platform 4.3 及更高版本中引进了性能上的改进，现在不再需要调整 **iptablesSyncPeriod** 参数。

字段	类型	描述
proxyArguments.iptables-min-sync-period	**数组**	刷新 **iptables** 规则前的最短时长。此字段确保刷新的频率不会过于频繁。有效的后缀包括 **s**、**m** 和 **h**，具体参见 Go **time** 软件包。默认值为： kubeProxyConfig: proxyArguments: iptables-min-sync-period: - 0s

5.2.12. 创建 Ignition 配置文件

由于需要手工启动集群机器，因此您必须生成 Ignition 配置文件，集群需要它来创建其机器。



重要

- 安装程序生成的 Ignition 配置文件包含在 24 小时后过期的证书，然后在过期时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外情况是，您需要手动批准待处理的 **node-bootstrapper** 证书签名请求（CSR）来恢复 kubelet 证书。如需更多信息，请参阅 *从过期的 control plane 证书中恢复*的文档。
- 建议您在生成 12 小时后使用 Ignition 配置文件，因为集群安装后 24 小时证书从 16 小时轮转至 22 小时。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中运行证书更新时避免安装失败。

先决条件

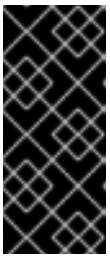
- 获取 OpenShift Container Platform 安装程序以及集群的 pull secret。

流程

- 获取 Ignition 配置文件：

```
$ ./openshift-install create ignition-configs --dir <installation_directory> 1
```

1 对于 **<installation_directory>**，请指定用于保存安装程序所创建的文件的目录名称。



重要

如果您创建了 **install-config.yaml** 文件，请指定包含该文件的目录。否则，指定一个空目录。一些安装信息，如 bootstrap X.509 证书，有较短的过期间隔，因此不要重复使用安装目录。如果要重复使用另一个集群安装中的个别文件，可以将其复制到您的目录中。但是，一些安装数据的文件名可能会在发行版本之间有所改变。从 OpenShift Container Platform 老版本中复制安装文件时要格外小心。

该目录中将生成以下文件：

```
.
```

```
├── auth
│   ├── kubeadmin-password
│   └── kubeconfig
├── bootstrap.ign
├── master.ign
├── metadata.json
└── worker.ign
```

5.2.13. 安装 RHCOS 并启动 OpenShift Container Platform bootstrap 过程

要在您置备的裸机基础架构上安装 OpenShift Container Platform，您必须在机器上安装 Red Hat Enterprise Linux CoreOS（RHCOS）。安装 RHCOS 时，您必须为 OpenShift Container Platform 安装程序生成的机器类型提供 Ignition 配置文件。如果您配置了合适的网络、DNS 和负载均衡基础架构，OpenShift Container Platform bootstrap 过程会在 RHCOS 机器重启后自动开始。

要在机器上安装 RHCOS，请按照以下步骤使用 ISO 镜像或网络 PXE 启动。

注意

本安装文档中包括的计算节点部署步骤特定于 RHCOS。如果您选择部署基于 RHEL 的计算节点，您将接管所有操作系统生命周期管理和维护，包括执行系统更新、应用补丁和完成所有其他必要的任务。RHEL 7 计算机器的使用已弃用，计划在以后的 OpenShift Container Platform 4 发行版本中删除。

您可以使用以下方法在 ISO 和 PXE 安装过程中配置 RHCOS:

- 内核参数： 您可以使用内核参数来提供特定于安装的信息。例如，您可以指定上传到 HTTP 服务器的 RHCOS 安装文件的位置，以及您要安装的节点类型的 Ignition 配置文件的位置。对于 PXE 安装，您可以使用 **APPEND** 参数将参数传递给实时安装程序的内核。对于 ISO 安装，您可以中断实时安装引导过程来添加内核参数。在这两种安装情况下，您可以使用特殊的 **coreos.inst.*** 参数来指示实时安装程序，以及标准安装引导参数来打开或关闭标准内核服务。

- Ignition 配置：OpenShift Container Platform Ignition 配置文件（***.ign**）特定于您要安装的节点类型。您可以在 RHCOS 安装过程中传递 bootstrap、control plane 或计算节点 Ignition 配置文件的位置，以便在第一次引导时生效。特殊情况下，您可以创建单独的、有限的 Ignition 配置来传递给 Live 系统。该 Ignition 配置可以执行特定任务，如在安装完成后向置备系统报告成功。这个特殊 Ignition 配置由 **coreos-installer** 使用，用于首次启动安装的系统。不要直接向 live ISO 提供标准 control plane 和计算节点 Ignition 配置。

- **coreos-installer**：您可以将 live ISO 安装程序引导到 shell 提示符，这可让您在首次引导前以多种方式准备持久性系统。特别是，您可以运行 **coreos-installer** 命令来识别包括的工件、使用磁盘分区以及设置联网。在有些情况下，您可以配置 live 系统上的功能并将其复制到安装的系统中。

使用 ISO 安装还是 PXE 安装要根据您的具体情况而定。PXE 安装需要可用的 DHCP 服务并进行更多准备，但可以使安装过程更自动化。ISO 安装是一个更手动的过程，如果您设置的机器较多，则可能不方便。

注意

自 OpenShift Container Platform 4.6 起，RHCOS ISO 和其他安装工件支持在带有 4K 扇区的磁盘上安装。

5.2.13.1. 使用 ISO 镜像创建 Red Hat Enterprise Linux CoreOS (RHCOS) 机器

在您置备的基础架构上安装集群前，必须先创建 RHCOS 机器供其使用。您可以使用 ISO 镜像来创建这些机器。

先决条件

- 获取集群的 Ignition 配置文件。
- 具有可从计算机以及您创建的机器访问的 HTTP 服务器的访问权限。

流程

1. 将安装程序创建的 control plane、计算和 bootstrap Ignition 配置文件上传到 HTTP 服务器。记下这些文件的 URL。

重要

如果您计划在安装完成后在集群中添加更多计算机器，请不要删除这些文件。

2. 从 RHCOS 镜像镜像 页面获取您选择的操作系统实例安装方法所需的 RHCOS 镜像。

重要

RHCOS 镜像可能不会随着 OpenShift Container Platform 的每一发行版本都有改变。您必须下载最高版本的镜像，其版本号应小于或等于您安装的 OpenShift Container Platform 版本。如果可用，请使用与 OpenShift Container Platform 版本匹配的镜像版本。此流程只使用 ISO 镜像。此安装类型不支持 RHCOS qcow2 镜像。

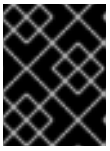
ISO 文件名类似以下示例：

rhcos-<version>-live.<architecture>.iso

3. 使用 ISO 启动 RHCOS 安装。使用如下安装选项之一：
   - 将 ISO 镜像刻录到磁盘并直接启动。
   - 通过 LOM 接口使用 ISO 重定向。
4. 引导 ISO 镜像。您可以中断安装引导过程来添加内核参数。然而，在这个 ISO 过程中，您应该使用 **coreos-installer** 命令而不是添加内核参数。如果您在没有选项或中断的情况下运行 live 安装程序，安装程序将引导至 live 系统上的 shell 提示符，准备好将 RHCOS 安装到磁盘中。
5. 在运行 **coreos-installer** 前，请参阅 *高级 RHCOS 安装参考* 部分，以了解配置功能的不同方法，如网络和磁盘分区。
6. 运行 **coreos-installer** 命令。您至少必须识别节点类型的 Ignition 配置文件位置，以及您要安装到的磁盘位置。下面是一个示例：

```
$ sudo coreos-installer install \
    --ignition-url=https://host/worker.ign /dev/sda
```

7. 安装 RHCOS 后，系统会重启。系统重启过程中，它会应用您指定的 Ignition 配置文件。
8. 继续为集群创建其他机器。



重要

此刻您必须创建 bootstrap 和 control plane 机器。如果 control plane 机器不可调度（这是默认调度），则在安装集群前至少会创建两台计算机器。

5.2.13.2. 通过 PXE 或 iPXE 启动来创建 Red Hat Enterprise Linux CoreOS (RHCOS) 机器

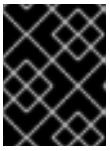
在安装使用手动置备 RHCOS 节点（如裸机）的集群前，您必须创建 RHCOS 机器供其使用。您可以使用 PXE 或 iPXE 启动来创建机器。

先决条件

- 获取集群的 Ignition 配置文件。
- 配置合适的 PXE 或 iPXE 基础架构。
- 具有 HTTP 服务器的访问权限，以便您可从计算机进行访问。

流程

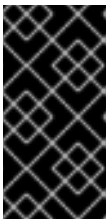
1. 将安装程序创建的 master、worker 和 bootstrap Ignition 配置文件上传到 HTTP 服务器。记下这些文件的 URL。



重要

您可以在 Ignition 配置中添加或更改配置设置，然后将其保存到 HTTP 服务器。如果您计划在安装完成后在集群中添加更多计算机器，请不要删除这些文件。

2. 从 RHCOS 镜像 镜像页面获取 RHCOS **内核、initram fs 和 rootfs** 文件。



重要

RHCOS 工件（artifact）可能不会随着 OpenShift Container Platform 的每个发行版本而改变。您必须下载最高版本的工件，其版本号应小于或等于您安装的 OpenShift Container Platform 版本。这个过程只使用下面描述的正确 **kernel**、**initramfs** 和 **rootfs** 工件。此安装类型不支持 RHCOS qcow2 镜像。

文件名包含 OpenShift Container Platform 版本号。它们类似以下示例：

- **kernel**: **rhcos-<version>-live-kernel-<architecture>**
- **initramfs**: **rhcos-<version>-live-initramfs.<architecture>.img**
- **rootfs**: **rhcos-<version>-live-rootfs.<architecture>.img**

3. 上传引导方法所需的额外文件：
   - 对于传统的 PXE，将 **kernel** 和 **initramfs** 文件上传到 TFTP 服务器，并将 **rootfs** 文件上传到 HTTP 服务器。
   - 对于 iPXE，将 **kernel**、**initram** fs 和 **rootfs** 文件上传到 HTTP 服务器。

重要

如果您计划在安装完成后在集群中添加更多计算机器，请不要删除这些文件。

4. 配置网络启动基础架构，以便在安装 RHCOS 后机器可从本地磁盘启动。

5. 为 RHCOS 镜像配置 PXE 或 iPXE 安装。
针对您的环境修改以下示例菜单条目之一，并验证能否正确访问镜像和 Ignition 文件：

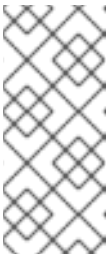
- 对于 PXE：

```
DEFAULT pxeboot
TIMEOUT 20
PROMPT 0
LABEL pxeboot
    KERNEL http://<HTTP_server>/rhcos-<version>-live-kernel-<architecture> (1)
    APPEND initrd=http://<HTTP_server>/rhcos-<version>-live-initramfs.
<architecture>.img coreos.live.rootfs_url=http://<HTTP_server>/rhcos-<version>-live-
rootfs.<architecture>.img coreos.inst.install_dev=/dev/sda
coreos.inst.ignition_url=http://<HTTP_server>/bootstrap.ign (2) (3)
```

(1) 指定上传到 HTTP 服务器的 live **kernel** 文件位置。URL 必须是 HTTP、TFTP 或者 FTP ; 不支持 HTTPS 和 NFS。

(2) 如果您使用多个 NIC，请在 **ip** 选项中指定一个接口。例如，要在名为 **eno1** 的 NIC 上使用 DHCP，请设置 **ip=eno1:dhcp**。

(3) 指定上传到 HTTP 服务器的 RHCOS 文件的位置。**initrd** 参数值是 **initramfs** 文件的位置，**coreos.live.rootfs_url** 参数值是 **rootfs** 文件的位置，**coreos.inst.ignition_url** 参数值是 bootstrap Ignition 配置文件的位置。您还可以在 **APPEND** 行中添加更多内核参数来配置联网或其他引导选项。



注意

这个配置不会在使用图形控制台的机器上启用串口控制台访问。要配置不同的控制台，请在 **APPEND** 行中添加一个或多个 **console=** 参数。例如，添加 **console=tty0 console=ttyS0** 将第一个 PC 串口设置为主控制台，图形控制台作为二级控制台。如需更多信息，请参阅如何在 Red Hat Enterprise Linux 中设置串行终端和（或）控制台？

- 对于 iPXE：

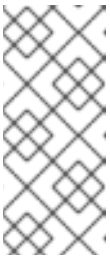
```
kernel http://<HTTP_server>/rhcos-<version>-live-kernel-<architecture> initrd=main
coreos.live.rootfs_url=http://<HTTP_server>/rhcos-<version>-live-rootfs.
<architecture>.img coreos.inst.install_dev=/dev/sda
coreos.inst.ignition_url=http://<HTTP_server>/bootstrap.ign (1) (2)
initrd --name main http://<HTTP_server>/rhcos-<version>-live-initramfs.
<architecture>.img (3)
boot
```

(1) 指定上传到 HTTP 服务器的 RHCOS 文件的位置。**kernel** 参数值是 **kernel** 文件的位置，在 UEFI 系统中引导时需要 **initrd=main** 参数。 **coreos.live.rootfs_url** 参数值是 **rootfs** 文件的位置， **coreos.inst.ignition_url** 参数值则是 bootstrap Ignition 配置文件

的位置。

2 如果您使用多个 NIC，请在 **ip** 选项中指定一个接口。例如，要在名为 **eno1** 的 NIC 上使用 DHCP，请设置 **ip=eno1:dhcp**。

3 指定上传到 HTTP 服务器的 **initramfs** 文件的位置。



注意

这个配置不会在使用图形控制台的机器上启用串口控制台访问。要配置不同的控制台，请在 **kerne** 行中添加一个或多个 **console=** 参数。例如，添加 **console=tty0 console=ttyS0** 将第一个 PC 串口设置为主控制台，图形控制台作为二级控制台。如需更多信息，请参阅如何在 Red Hat Enterprise Linux 中设置串行终端和（或）控制台？

6. 如果使用 PXE UEFI，请执行以下操作：

   a. 提供引导系统所需的 **shim x64.efi 和 grubx64 .efi EFI 二进制文件以及 grub.cfg** 文件。

   - 通过将 RHCOS ISO 挂载到主机，然后将 **images/efiboot.img** 文件挂载到您的主机来提取所需的 EFI 二进制文件：

     ```
     $ mkdir -p /mnt/iso
     ```

     ```
     $ mkdir -p /mnt/efiboot
     ```

     ```
     $ mount -o loop rhcos-installer.x86_64.iso /mnt/iso
     ```

     ```
     $ mount -o loop,ro /mnt/iso/images/efiboot.img /mnt/efiboot
     ```

   - 从 **efiboot.img** 挂载点，将 **EFI/redhat/shimx64.efi** 和 **EFI/redhat/grubx64.efi** 文件复制到 TFTP 服务器中：

     ```
     $ cp /mnt/efiboot/EFI/redhat/shimx64.efi .
     ```

     ```
     $ cp /mnt/efiboot/EFI/redhat/grubx64.efi .
     ```

     ```
     $ umount /mnt/efiboot
     ```

     ```
     $ umount /mnt/iso
     ```

   - 将 RHCOS ISO 中包含的 **EFI/redhat/grub.cfg** 文件复制到您的 TFTP 服务器中。

   b. 编辑 **grub.cfg** 文件使其包含类似如下的参数：

   ```
   menuentry 'Install Red Hat Enterprise Linux CoreOS' --class fedora --class gnu-linux --class gnu --class os {
    linuxefi rhcos-<version>-live-kernel-<architecture> coreos.inst.install_dev=/dev/sda coreos.live.rootfs_url=http://<HTTP_server>/rhcos-<version>-live-rootfs.
   ```

```
<architecture>.img coreos.inst.ignition_url=http://<HTTP_server>/bootstrap.ign
 initrdefi rhcos-<version>-live-initramfs.<architecture>.img
}
```

其中：

rhcos-<version>-live-kernel-<architecture>

指定上传到 TFTP 服务器的 **内核** 文件。

http://<HTTP_server>/rhcos-<version>-live-rootfs.<architecture>.img

指定上传到 HTTP 服务器的 live rootfs 镜像的位置。

http://<HTTP_server>/bootstrap.ign

指定上传到 HTTP 服务器的 bootstrap Ignition 配置文件的位置。

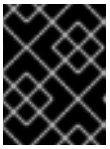
rhcos-<version>-live-initramfs.<architecture>.img

指定上传到 TFTP 服务器的 **initramfs** 文件的位置。

注意

有关如何为 UEFI 引导配置 PXE 服务器的更多信息，请参阅红帽知识库文章：如何为 Red Hat Enterprise Linux 的 UEFI 引导配置/设置 PXE 服务器？

7. 继续为集群创建机器。



重要

此刻您必须创建 bootstrap 和 control plane 机器。如果 control plane 机器不可调度（这是默认调度），则在安装集群前至少会创建两台计算机器。

5.2.13.3. 高级 Red Hat Enterprise Linux CoreOS（RHCOS）安装配置

为 OpenShift Container Platform 手动置备 Red Hat Enterprise Linux CoreOS（RHCOS）节点的一个关键优点是能够进行通过默认的 OpenShift Container Platform 安装方法无法进行的配置。本节介绍了您可以使用的一些技术来进行配置，其中包括：

- 将内核参数传递给实时安装程序
- 从 live 系统手动运行 **coreos-installer**
- 将 Ignition 配置嵌入 ISO 中

本节详述了与 Red Hat Enterprise Linux CoreOS（RHCOS）手动安装的高级配置相关的内容，如磁盘分区、网络以及使用 Ignition 配置的不同方式相关。

5.2.13.3.1. 使用高级网络选项进行 PXE 和 ISO 安装

OpenShift Container Platform 节点的网络默认使用 DHCP 来收集所有必要配置设置。要设置静态 IP 地址或配置特殊的设置，如绑定,您可以执行以下操作之一：

- 引导 live 安装程序时会传递特殊的内核参数。
- 使用机器配置将网络文件复制到安装的系统中。

- 使用 live installer shell 提示配置网络，然后将那些设置复制到安装的系统上，以便在安装的系统第一次引导时生效。

要配置 PXE 或 iPXE 安装，请使用以下选项之一：

- 请参阅"高级 RHCOS 安装参考"表。
- 使用机器配置将网络文件复制到安装的系统中。

要配置 ISO 安装，请使用以下步骤。

流程

1. 引导 ISO 安装程序。
2. 在 live 系统 shell 提示下，使用可用的 RHEL 工具（如 **nmcli** 或 **nmtui**）为 Live 系统配置网络。
3. 运行 **coreos-installer** 命令来安装系统，添加 **--copy-network** 选项来复制网络配置。例如：

```
$ coreos-installer install --copy-network \
    --ignition-url=http://host/worker.ign /dev/sda
```

重要

copy-network 选项只复制 **/etc/NetworkManager/system-connections** 下的网络配置。特别是，它不会复制系统主机名。

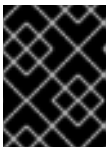
4. 重启安装的系统。

5.2.13.3.2. 磁盘分区

磁盘分区是在 Red Hat Enterprise Linux CoreOS（RHCOS）安装过程中在 OpenShift Container Platform 集群节点上创建的。特定架构的每个 RHCOS 节点都使用相同的分区布局，除非默认分区配置被覆盖。在 RHCOS 安装过程中，根文件系统的大小会增大，以使用目标设备中剩余的可用空间。

但是，在安装 OpenShift Container Platform 节点时，在两种情况下您可能需要覆盖默认分区：

- 创建单独的分区：对于在空磁盘中的 greenfield 安装，您可能想要在分区中添加单独的存储。这只在生成 **/var** 或者一个 **/var** 独立分区的子目录（如 **/var/lib/etcd** ）时被正式支持，但不支持两者。



重要

Kubernetes 只支持两个文件系统分区。如果您在原始配置中添加多个分区，Kubernetes 无法监控所有这些分区。

- 保留现有分区：对于 brownfield 安装，您要在现有节点上重新安装 OpenShift Container Platform，并希望保留从之前的操作系统中安装的数据分区，对于 **coreos-installer** 来说，引导选项和选项都允许您保留现有数据分区。

5.2.13.3.2.1. 创建一个独立的 /var 分区

通常情况下，OpenShift Container Platform 的磁盘分区应该留给安装程序。然而，在有些情况下您可能需要在文件系统的一部分中创建独立分区。

OpenShift Container Platform 支持添加单个分区来将存储附加到 **/var** 分区或 **/var** 的子目录。例如：

- **/var/lib/containers**：保存镜像相关的内容，随着更多镜像和容器添加到系统中，它所占用的存储会增加。
- **/var/lib/etcd**：保存您可能希望保持独立的数据，比如 etcd 存储的性能优化。
- **/var**：保存您希望独立保留的数据，用于特定目的（如审计）。

单独存储 **/var** 目录的内容可方便地根据需要对区域扩展存储，并可以在以后重新安装 OpenShift Container Platform 时保持该数据地完整。使用这个方法，您不必再次拉取所有容器，在更新系统时也无法复制大量日志文件。

因为 **/var** 在进行一个全新的 Red Hat Enterprise Linux CoreOS（RHCOS）安装前必需存在，所以这个流程会在 OpenShift Container Platform 安装过程的 **openshift-install** 准备阶段插入的机器配置来设置独立的 **/var** 分区。

流程

1. 创建存放 OpenShift Container Platform 安装文件的目录：

```
$ mkdir $HOME/clusterconfig
```

2. 运行 **openshift-install** 在 **manifest** 和 **openshift** 子目录中创建一组文件。在出现提示时回答系统问题：

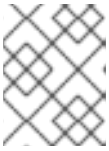
```
$ openshift-install create manifests --dir $HOME/clusterconfig
? SSH Public Key ...
$ ls $HOME/clusterconfig/openshift/
99_kubeadmin-password-secret.yaml
99_openshift-cluster-api_master-machines-0.yaml
99_openshift-cluster-api_master-machines-1.yaml
99_openshift-cluster-api_master-machines-2.yaml
...
```

3. 创建 **MachineConfig** 对象并将其添加到 **openshift** 目录中的一个文件中。例如，把文件命名为 **98-var-partition.yaml**，将磁盘设备名称改为 **worker** 系统中存储设备的名称，并根据情况设置存储大小。这个示例将 **/var** 目录放在一个单独的分区中：

```
apiVersion: machineconfiguration.openshift.io/v1
kind: MachineConfig
metadata:
  labels:
    machineconfiguration.openshift.io/role: worker
  name: 98-var-partition
spec:
  config:
    ignition:
      version: 3.1.0
    storage:
      disks:
      - device: /dev/<device_name> 1
        partitions:
        - label: var
          startMiB: <partition_start_offset> 2
```

```
      sizeMiB: <partition_size> 3
  filesystems:
    - device: /dev/disk/by-partlabel/var
      path: /var
      format: xfs
systemd:
  units:
    - name: var.mount 4
      enabled: true
      contents: |
        [Unit]
        Before=local-fs.target
        [Mount]
        What=/dev/disk/by-partlabel/var
        Where=/var
        Options=defaults,prjquota 5
        [Install]
        WantedBy=local-fs.target
```

1. 要分区的磁盘的存储设备名称。
2. 当在引导磁盘中添加数据分区时，推荐最少使用 25000MB。root 文件系统会自动重新定义大小使其占据所有可用空间（最多到指定的偏移值）。如果没有指定值，或者指定的值小于推荐的最小值，则生成的 root 文件系统会太小，而在以后进行的 RHCOS 重新安装可能会覆盖数据分区的开始部分。
3. 数据分区的大小（以兆字节为单位）。
4. 挂载单元的名称必须与 **Where=** 指令中指定的目录匹配。例如，对于挂载在 **/var/lib/containers** 上的文件系统，该单元必须命名为 **var-lib-containers.mount**。
5. 对于用于容器存储的文件系统，必须启用 **prjquota** 挂载选项。



注意

在创建独立 **/var** 分区时，如果不同的实例类型没有相同的设备名称，则无法将不同的实例类型用于 worker 节点。

4. 再次运行 **openshift-install**，从 **manifest** 和 **openshift** 子目录中的一组文件创建 Ignition 配置：

```
$ openshift-install create ignition-configs --dir $HOME/clusterconfig
$ ls $HOME/clusterconfig/
auth  bootstrap.ign  master.ign  metadata.json  worker.ign
```

现在，可以使用 Ignition 配置文件作为 ISO 或 PXE 手动安装过程的输入来安装 Red Hat Enterprise Linux CoreOS（RHCOS）系统。

5.2.13.3.2.2. 保留现有分区

对于 ISO 安装，您可以在 **coreos-installer** 命令行中添加可让安装程序维护一个或多个现有分区的选项。对于 PXE 安装，您可以 **APPEND coreos.inst.*** 选项来保留分区。

保存的分区可能是来自现有 OpenShift Container Platform 系统中的分区，其中包括了您希望保留的数据分区。以下是几个提示：

- 如果您保存了现有分区，且这些分区没有为 RHCOS 留下足够空间，则安装将失败但不会损害已保存的分区。
- 通过分区标签或数字识别您要保留的磁盘分区。

对于 ISO 安装

这个示例保留分区标签以**数据（data***）开头的任何分区：

```
# coreos-installer install --ignition-url http://10.0.2.2:8080/user.ign \
      --save-partlabel 'data*' /dev/sda
```

以下示例演示了在运行 **coreos-installer** 时要保留磁盘上的第 6 个分区：

```
# coreos-installer install --ignition-url http://10.0.2.2:8080/user.ign \
      --save-partindex 6 /dev/sda
```

这个示例保留了分区 5 及更高分区：

```
# coreos-installer install --ignition-url http://10.0.2.2:8080/user.ign
      --save-partindex 5- /dev/sda
```

在前面已保存分区的示例中，**coreos-installer** 会立即重新创建分区。

对于 PXE 安装

这个 **APPEND** 选项保留分区标签以 'data'('data*')开头的所有分区：

```
coreos.inst.save_partlabel=data*
```

这个 **APPEND** 选项保留分区 5 及其后的分区：

```
coreos.inst.save_partindex=5-
```

这个 **APPEND** 选项保留分区 6:

```
coreos.inst.save_partindex=6
```

5.2.13.3.3. 标识 Ignition 配置

在进行 RHCOS 手动安装时，您可以提供两种 Ignition 配置类型，它们有不同的原因：

- **永久安装 Ignition 配置**：每个手动 RHCOS 安装都需要传递 **openshift-installer** 生成的 Ignition 配置文件之一，如 **bootstrap.ign**、**master.ign** 和 **worker.ign**，才能进行安装。

  重要

  不建议修改这些文件。

  对于 PXE 安装，您可以使用 **coreos.inst.ignition_url=** 选项在 **APPEND** 行上传递 Ignition 配置。对于 ISO 安装，在 ISO 引导至 shell 提示符后，您可以使用 **--ignition-url=** 选项在 **coreos-installer** 命令行上识别 Ignition 配置。在这两种情况下，都只支持 HTTP 和 HTTPS 协议。

- **live 安装 Ignition 配置**：此类型必须手动创建，并应该尽可能避免，因为红帽不支持它。使用此方法，Ignition 配置会传递到 live 安装介质，在引导时立即运行，并在 RHCOS 系统安装到磁盘之前和/或之后执行设置任务。这个方法只用于必须执行一次且之后不能再次应用的任务，如不能使用机器配置进行的高级分区。
  对于 PXE 或 ISO 引导，您可以创建 Ignition 配置，**APPEND ignition.config.url=** 选项，以标识 Ignition 配置的位置。您还需要附加 **ignition.firstboot ignition.platform.id=metal** 或者 **ignition.config.url** 选项。

5.2.13.3.3.1. 在 RHCOS ISO 中嵌入 Ignition 配置

您可以直接嵌入 RHCOS ISO 镜像中的 live 安装 Ignition 配置。引导 ISO 镜像后，内嵌的配置将自动应用。

流程

1. 从以下镜像镜像页面下载 **coreos-installer** 二进制文件：
   https://mirror.openshift.com/pub/openshift-v4/clients/coreos-installer/latest/。
2. 检索 RHCOS ISO 镜像和 Ignition 配置文件，并将其复制到可访问的目录中，如 **/mnt**:

```
# cp rhcos-<version>-live.x86_64.iso bootstrap.ign /mnt/
# chmod 644 /mnt/rhcos-<version>-live.x86_64.iso
```

3. 运行以下命令将 Ignition 配置嵌入 ISO 中：

```
# ./coreos-installer iso ignition embed -i /mnt/bootstrap.ign \
    /mnt/rhcos-<version>-live.x86_64.iso
```

   现在，您以使用该 ISO 使用指定的 live 安装 Ignition 配置来安装 RHCOS。

重要

不支持且不推荐使用 **coreos-installer iso ignition embed** 来嵌入由 **openshift-installer** 生成的文件，如 **bootstrap.ign**、**master.ign** 和 **worker.ign**。

4. 要显示嵌入的 Ignition 配置的内容并将其定向到文件中，请运行：

```
# ./coreos-installer iso ignition show /mnt/rhcos-<version>-live.x86_64.iso > mybootstrap.ign
```

```
# diff -s bootstrap.ign mybootstrap.ign
```

输出示例

```
Files bootstrap.ign and mybootstrap.ign are identical
```

5. 要删除 Ignition 配置并将 ISO 返回到其 pristine 状态（因此您可以重复使用它），请运行：

```
# ./coreos-installer iso ignition remove /mnt/rhcos-<version>-live.x86_64.iso
```

   现在，您可以将另一个 Ignition 配置嵌入到 ISO 中，或者在其 pristine 状态下使用 ISO。

5.2.13.3.4. 高级 RHCOS 安装参考

本节演示了网络配置和其他高级选项，允许您修改 Red Hat Enterprise Linux CoreOS（RHCOS）手动安装过程。下表描述了您可以与 RHCOS live installer 和 **coreos-installer** 命令一起使用的内核参数和命令行选项。

RHCOS 启动提示下的路由和绑定选项
如果从 ISO 镜像安装 RHCOS，您可以在引导该镜像时手动添加内核参数以配置节点的网络。如果没有使用网络参数，则安装默认为使用 DHCP。

重要

添加网络参数时，还必须添加 **rd.neednet=1** 内核参数。

下表描述了如何为实时 ISO 安装使用 **ip=** 、 **nameserver=** 和 **bond=** 内核参数。

注意

在添加内核参数时顺序非常重要： **ip=**，**nameserver=**，然后 **bond=**。

ISO 的路由和绑定选项

下表提供了配置 Red Hat Enterprise Linux CoreOS（RHCOS）节点网络的示例。这些是在系统引导过程中传递给 **dracut** 工具的网络选项。有关 **dracut** 支持的网络选项的详情，请参考 **dracut.cmdline** 手册页。

描述	示例
要配置一个 IP 地址，可以使用 DHCP(**ip=dhcp**)或者设置单独的静态 IP 地址(**ip=<host_ip>**)。然后在每个节点上指定 DNS 服务器 IP 地址(**nameserver=<dns_ip>**)。这个示例设置： • 节点的 IP 地址为 **10.10.10.2** • 网关地址为 **10.10.10.254** • 子网掩码为 **255.255.255.0** • 主机名为 **core0.example.com** • DNS 服务器地址为 **4.4.4.41**	ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp1s0:none nameserver=4.4.4.41
通过指定多个 **ip=** 条目来指定多个网络接口。	ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp1s0:none ip=10.10.10.3::10.10.10.254:255.255.255.0:core0.example.com:enp2s0:none

描述	示例

可选：您可以通过设置一个 **rd.route=** 值来配置到额外网络的路由。
 	
如果额外网络网关与主要网络网关不同，则默认网关必须是主要网络网关。
	配置默认网关：
	
	`ip=::10.10.10.254::::`
	
	为额外网络配置路由：
	
	`rd.route=20.20.20.0/24:20.20.20.254:enp2s0`

在单一接口中禁用 DHCP，比如当有两个或者多个网络接口时，且只有一个接口被使用。	`ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp1s0:none` `ip=::::core0.example.com:enp2s0:none`
您可以将系统中 DHCP 和静态 IP 配置与多个网络接口结合在一起。	`ip=enp1s0:dhcp` `ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp2s0:none`

可选： 您可以使用 **vlan=** 参数在单独的接口上配置 VLAN。	在网络接口中配置 VLAN 并使用静态 IP 地址：
	
	`ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp2s0.100:none`
	`vlan=enp2s0.100:enp2s0`
	
	在网络接口中配置 VLAN 并使用 DHCP：
	
	`ip=enp2s0.100:dhcp`
	`vlan=enp2s0.100:enp2s0`

您可以为每个服务器添加一个 **nameserver=** 条目来提供多个 DNS 服务器。	`nameserver=1.1.1.1` `nameserver=8.8.8.8`

可选：使用 **bond=** 选项支持将多个网络接口绑定到一个接口。在这两个示例中：
 	
• 配置绑定接口的语法为：**bond=name[:network_interfaces][:options]**
• *name* 是绑定设备名称（**bond0**），*network_interfaces* 代表用逗号分开的物理（以太网）接口(**em1,em2**)的列表，*options* 是用逗号分开的绑定选项列表。输入 **modinfo bonding** 查看可用选项。
• 当使用 **bond=** 创建绑定接口时，您必须指定如何分配 IP 地址以及绑定接口的其他信息。
	要将绑定的接口配置为使用 DHCP，请将绑定的 IP 地址设置为 **dhcp**。例如：
	
	`bond=bond0:em1,em2:mode=active-backup`
	`ip=bond0:dhcp`
	
	要将绑定接口配置为使用静态 IP 地址，请输入您需要的特定 IP 地址以及相关信息。例如：
	
	`bond=bond0:em1,em2:mode=active-backup`
	`ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:bond0:none`

描述	示例
可选： 您可以使用 **vlan=** 参数在绑定接口上配置 VLAN。	使用 VLAN 配置绑定接口并使用 DHCP： ip=bond0.100:dhcp bond=bond0:em1,em2:mode=active-backup vlan=bond0.100:bond0 使用 VLAN 配置绑定接口，并使用静态 IP 地址： ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:bond0.100:none bond=bond0:em1,em2:mode=active-backup vlan=bond0.100:bond0
可选：通过使用 **team=** 参数，网络合作可用作绑定的替代选择。在此例中： • 配置组接口的语法为：**team=name[:network_interfaces]** *name* 是组设备名称(**team0**)，**network_interfaces** 代表以逗号分隔的物理（以太网）接口（**em1**、**em2**）列表。 **注意** 当 RHCOS 切换到即将推出的 RHEL 版本时，团队计划会被弃用。如需更多信息，请参阅红帽知识库文章。	配置网络团队： team=team0:em1,em2 ip=team0:dhcp

coreos.inst 引导选项用于 ISO 或 PXE 安装

虽然您可以将大多数标准安装引导参数传递给 live 安装程序，但也有一些特定于 RHCOS live 安装程序的参数。

- 对于 ISO，可以通过中断 RHCOS 安装程序来添加这些选项。
- 对于 PXE 或 iPXE，这些选项必须在启动 PXE 内核前添加到 **APPEND** 行中。您无法中断实时 PXE 安装。

下表显示了用于 ISO 和 PXE 安装的 RHCOS live installer 引导选项。

表 5.28. coreos.inst 引导选项

参数	描述
coreos.inst.install_dev	必需。要安装的系统中的块设备。虽然可以使用 **sda** 这样的相对路径，但建议使用完整路径，如 **/dev/sda**。

参数	描述
coreos.inst.ignition_url	可选：嵌入到已安装系统中的 Ignition 配置的 UR如果没有指定 URL，则不会嵌入 Ignition 配置。
coreos.inst.save_partlabel	可选：在安装过程中要保留的分区压缩标签。允许使用 glob 风格的通配符。指定分区不需要存在。
coreos.inst.save_partindex	可选：在安装过程中完成要保留的分区的分离索引。可以使用 **m-n** 指定范围，**m** 或 **n** 可以被省略。指定分区不需要存在。
coreos.inst.insecure	可选：将 **coreos.inst.image_url** 指定的 OS 镜像提交取消签名。
coreos.inst.image_url	可选：下载并安装指定的 RHCOS 镜像。 ● 这个参数不应该在生产环境中使用，而是只用于调试目的。 ● 虽然在 RHCOS 的安装版本与 live 介质的版本不匹配时可以使用这个参数，但建议使用与您要安装版本匹配的介质。 ● 如果您使用的是 **coreos.inst.image_url**。还必须使用 **coreos.inst.insecure**。这是因为，裸机介质没有为 OpenShift Container Platform 进行 GPG 签名。 ● 只支持 HTTP 和 HTTPS 协议。
coreos.inst.skip_reboot	可选：安装后该系统不会重启。安装完成后，您会收到提示，提示您检查在安装过程中发生的情况。这个参数不应该在生产环境中使用，而是只用于调试目的。
coreos.inst.platform_id	可选：安装 RHCOS 镜像的平台的 Ignition 平台 ID。默认为 **metal**。这个选项决定是否从云供应商（如 VMware）请求 Ignition 配置。例如：**coreos.inst.platform_id=vmware**。
ignition.config.url	可选：用于实时启动的 Ignition 配置的 URL。例如，它可以用来定制调用 **coreos-installer** 的方式，或者用来在安装前或安装后运行代码。这与 **coreos.inst.ignition_url**（这是已安装系统的 Ignition 配置）不同。

ISO 安装的 coreos-installer 选项

您还可以直接从命令行调用 **coreos-installer** 命令来安装 RHCOS。上表中的内核参数提供了在引导时自动调用 **coreos-installer** 的快捷方式，但您可以在 shell 提示符运行时将类似的参数直接传递给 **coreos-installer**。

下表显示了您可以在实时安装过程中从 shell 提示符传递给 **coreos-installer** 命令的选项和子命令。

表 5.29. CoreOS-installer 命令行选项、参数和子命令

命令行选项	
选项	**描述**
-u, **--image-url <url>**	手动指定镜像 URL。
-f, **--image-file <path>**	手动指定本地镜像文件。
-i, --ignition-file <path>	从文件中嵌入 Ignition 配置。
-I, **--ignition-url <URL>**	从 URL 嵌入 Ignition 配置。
--ignition-hash <digest>	Ignition config 的 **type-value** 的文摘值。
-p, **--platform <name>**	覆盖 Ignition 平台 ID。
--append-karg <arg>…	附加默认内核参数。
--delete-karg <arg>…	删除默认内核参数。
-n, **--copy-network**	从安装环境中复制网络配置。  **重要** **copy-network** 选项只复制 **/etc/NetworkManager/system-connections** 下的网络配置。特别是，它不会复制系统主机名。
--network-dir <path>	使用 **-n**。默认为 **/etc/NetworkManager/system-connections/**。
--save-partlabel <lx>..	使用这个标签 glob 保存分区。
--save-partindex <id>…	使用这个数值或者范围保存分区。
--offline	强制离线安装。
--insecure	跳过签名验证。
--insecure-ignition	允许没有 HTTPS 或 hash 的 Ignition URL。
--architecture <name>	目标 CPU 架构。默认为 **x86_64**。
--preserve-on-error	出现错误时不清除分区表。

-h，--help	打印帮助信息。
命令行参数	
参数	**描述**
<device>	目的设备。
CoreOS-installer 嵌入的 Ignition 命令	
命令	**描述**
$ coreos-installer iso ignition embed <options> --ignition-file <file_path> <ISO_image>	在 ISO 镜像中嵌入 Ignition 配置。
coreos-installer iso ignition show <options> <ISO_image>	显示来自 ISO 镜像的内嵌 Ignition 配置。
coreos-installer iso ignition remove <options> <ISO_image>	从 ISO 镜像中删除嵌入的 Ignition 配置。
coreos-installer ISO Ignition 选项	
选项	**描述**
-f, --force	覆盖现有的 Ignition 配置。
-i, --ignition-file <path>	要使用的 Ignition 配置。默认为 **stdin**。
-o, --output <path>	将 ISO 写入到一个新输出文件。
-h，--help	打印帮助信息。
coreos-installer PXE Ignition 命令	
命令	**描述**
请注意，不是所有子命令都接受这些选项。	
coreos-installer pxe ignition wrap <options>	在镜像中嵌套 Ignition 配置。
coreos-installer pxe ignition unwrap <options> <image_name>	显示在镜像中嵌套的 Ignition 配置。
coreos-installer pxe ignition unwrap <options> <initrd_name>	在 **initrd** 镜像中显示嵌套的 Ignition 配置。

coreos-installer PXE Ignition 选项	
选项	**描述**
-i, --ignition-file <path>	要使用的 Ignition 配置。默认为 **stdin**。
-o, --output <path>	将 ISO 写入到一个新输出文件。
-h, --help	打印帮助信息。

5.2.14. 创建集群

要创建 OpenShift Container Platform 集群，请等待您通过安装程序生成的 Ignition 配置文件所置备的机器上完成 bootstrap 过程。

先决条件

- 为集群创建所需的基础架构。
- 已获得安装程序并为集群生成了 Ignition 配置文件。
- 已使用 Ignition 配置文件为集群创建 RHCOS 机器。
- 您的机器可直接访问互联网，或者可以使用 HTTP 或 HTTPS 代理。

流程

1. 监控 bootstrap 过程：

```
$ ./openshift-install --dir <installation_directory> wait-for bootstrap-complete \ 1
    --log-level=info 2
```

1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

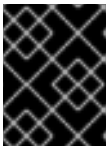
2 要查看不同的安装详情，请指定 **warn**、**debug** 或 **error**，而不要指定 **info**。

输出示例

```
INFO Waiting up to 30m0s for the Kubernetes API at https://api.test.example.com:6443...
INFO API v1.19.0 up
INFO Waiting up to 30m0s for bootstrapping to complete...
INFO It is now safe to remove the bootstrap resources
```

Kubernetes API 服务器提示已在 control plane 机器上完成 bootstrap 时，命令运行成功。

2. bootstrap 过程完成后，请从负载均衡器中删除 bootstrap 机器。



重要

此时您必须从负载均衡器中删除 bootstrap 机器。您还可以删除或重新格式化机器本身。

5.2.15. 使用 CLI 登录到集群

您可以通过导出集群 **kubeconfig** 文件，以默认系统用户身份登录集群。**kubeconfig** 文件包含关于集群的信息，供 CLI 用于将客户端连接到正确集群和 API 服务器。该文件特只适用于一个特定的集群，在 OpenShift Container Platform 安装过程中创建。

先决条件

- 已部署了 OpenShift Container Platform 集群。
- 已安装 **oc** CLI。

流程

1. 导出 **kubeadmin** 凭证：

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

2. 使用导出的配置，验证能否成功运行 **oc** 命令：

```
$ oc whoami
```

输出示例

```
system:admin
```

5.2.16. 批准机器的证书签名请求

将机器添加到集群时，会为您添加的每台机器生成两个待处理证书签名请求（CSR）。您必须确认这些 CSR 已获得批准，或根据需要自行批准。客户端请求必须首先被批准，然后是服务器请求。

先决条件

- 您已将机器添加到集群中。

流程

1. 确认集群可以识别这些机器：

```
$ oc get nodes
```

输出示例

```
NAME     STATUS   ROLES   AGE  VERSION
master-0 Ready    master  63m  v1.19.0
```

```
master-1  Ready    master  63m  v1.19.0
master-2  Ready    master  64m  v1.19.0
```

输出将列出您创建的所有机器。

注意

在一些 CSR 被批准前，以上输出可能不包括计算节点（也称为 worker 节点）。

2. 检查待处理的 CSR，并确保可以看到添加到集群中的每台机器都有 **Pending** 或 **Approved** 状态的客户端请求：

```
$ oc get csr
```

输出示例

```
NAME        AGE    REQUESTOR                                                   CONDITION
csr-8b2br   15m    system:serviceaccount:openshift-machine-config-operator:node-
bootstrapper   Pending
csr-8vnps   15m    system:serviceaccount:openshift-machine-config-operator:node-
bootstrapper   Pending
...
```

在本例中，两台机器加入了集群。您可能会在列表中看到更多已批准的 CSR。

3. 如果 CSR 没有获得批准，请在所添加机器的所有待处理 CSR 都处于 **Pending** 状态后，为您的集群机器批准这些 CSR：

注意

由于 CSR 会自动轮转，因此请在将机器添加到集群后一小时内批准您的 CSR。如果没有在一小时内批准，证书将会轮转，每个节点将会存在多个证书。您必须批准所有这些证书。批准客户端 CSR 后， Kubelet 为服务证书创建辅助 CSR，这需要手动批准。然后，如果 Kubelet 请求具有相同参数的新证书，则 **machine-approver** 会自动批准后续服务证书续订请求。

注意

对于在未启用机器 API 的平台中运行的集群，如裸机和其他用户置备的基础架构，必须采用一种方法自动批准 kubelet 提供证书请求（CSR）。如果没有批准请求，则 **oc exec**、**oc rsh** 和 **oc logs** 命令将无法成功，因为 API 服务器连接到 kubelet 时需要服务证书。与 Kubelet 端点联系的任何操作都需要此证书批准。这个方法必须监视新的 CSR，确认 CSR 由 **system:node** 或 **system:admin** 组中的 **node-bootstrapper** 服务帐户提交，并确认节点的身份。

- 若要单独批准，请对每个有效的 CSR 运行以下命令：

```
$ oc adm certificate approve <csr_name> 1
```

1 **<csr_name>** 是当前 CSR 列表中 CSR 的名称。

- 要批准所有待处理的 CSR，请运行以下命令：

```
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}{{"\n"}}
{{end}}{{end}}' | xargs --no-run-if-empty oc adm certificate approve
```

注意

在有些 CSR 被批准前，一些 Operator 可能无法使用。

4. 现在，您的客户端请求已被批准，您必须查看添加到集群中的每台机器的服务器请求：

```
$ oc get csr
```

输出示例

```
NAME        AGE     REQUESTOR                                                   CONDITION
csr-bfd72   5m26s   system:node:ip-10-0-50-126.us-east-2.compute.internal
Pending
csr-c57lv   5m26s   system:node:ip-10-0-95-157.us-east-2.compute.internal
Pending
...
```

5. 如果剩余的 CSR 没有被批准，且处于 **Pending** 状态，请批准集群机器的 CSR：

- 若要单独批准，请对每个有效的 CSR 运行以下命令：

```
$ oc adm certificate approve <csr_name> 1
```

1 **<csr_name>** 是当前 CSR 列表中 CSR 的名称。

- 要批准所有待处理的 CSR，请运行以下命令：

```
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}{{"\n"}}
{{end}}{{end}}' | xargs oc adm certificate approve
```

6. 批准所有客户端和服务器 CSR 后，器将处于 **Ready** 状态。运行以下命令验证：

```
$ oc get nodes
```

输出示例

```
NAME      STATUS    ROLES   AGE  VERSION
master-0  Ready     master  73m  v1.20.0
master-1  Ready     master  73m  v1.20.0
master-2  Ready     master  74m  v1.20.0
worker-0  Ready     worker  11m  v1.20.0
worker-1  Ready     worker  11m  v1.20.0
```

注意

批准服务器 CSR 后可能需要几分钟时间让机器转换为 **Ready** 状态。

其他信息

- 如需有关 CSR 的更多信息，请参阅证书签名请求。

5.2.17. 初始 Operator 配置

在 control plane 初始化后，您必须立即配置一些 Operator 以便它们都可用。

先决条件

- 您的 control plane 已初始化。

流程

1. 观察集群组件上线：

```
$ watch -n5 oc get clusteroperators
```

输出示例

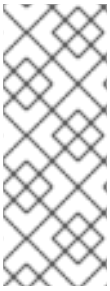
```
NAME                                       VERSION AVAILABLE   PROGRESSING   DEGRADED
SINCE
authentication                             4.6.0   True        False         False      3h56m
cloud-credential                           4.6.0   True        False         False      29h
cluster-autoscaler                         4.6.0   True        False         False      29h
config-operator                            4.6.0   True        False         False      6h39m
console                                    4.6.0   True        False         False      3h59m
csi-snapshot-controller                    4.6.0   True        False         False      4h12m
dns                                        4.6.0   True        False         False      4h15m
etcd                                       4.6.0   True        False         False      29h
image-registry                             4.6.0   True        False         False      3h59m
ingress                                    4.6.0   True        False         False      4h30m
insights                                   4.6.0   True        False         False      29h
kube-apiserver                             4.6.0   True        False         False      29h
kube-controller-manager                    4.6.0   True        False         False      29h
kube-scheduler                             4.6.0   True        False         False      29h
kube-storage-version-migrator              4.6.0   True        False         False      4h2m
machine-api                                4.6.0   True        False         False      29h
machine-approver                           4.6.0   True        False         False      6h34m
machine-config                             4.6.0   True        False         False      3h56m
marketplace                                4.6.0   True        False         False      4h2m
monitoring                                 4.6.0   True        False         False      6h31m
network                                    4.6.0   True        False         False      29h
node-tuning                                4.6.0   True        False         False      4h30m
openshift-apiserver                        4.6.0   True        False         False      3h56m
openshift-controller-manager               4.6.0   True        False         False      4h36m
openshift-samples                          4.6.0   True        False         False      4h30m
operator-lifecycle-manager                 4.6.0   True        False         False      29h
operator-lifecycle-manager-catalog         4.6.0   True        False         False      29h
operator-lifecycle-manager-packageserver   4.6.0   True        False         False      3h59m
service-ca                                 4.6.0   True        False         False      29h
storage                                    4.6.0   True        False         False      4h30m
```

2. 配置不可用的 Operator。

5.2.17.1. 安装过程中删除的镜像 registry

在不提供可共享对象存储的平台上，OpenShift Image Registry Operator bootstraps 本身的状态是 **Removed**。这允许 **openshift-installer** 在这些平台类型上完成安装。

将 **ManagementState** Image Registry Operator 配置从 **Removed** 改为 **Managed**。



注意

Prometheus 控制台提供了一个 **ImageRegistryRemoved** 警报，例如：

"Image Registry has been removed.**ImageStreamTags**, **BuildConfigs** and **DeploymentConfigs** which reference **ImageStreamTags** may not work as expected.Please configure storage and update the config to **Managed** state by editing configs.imageregistry.operator.openshift.io."

5.2.17.2. 镜像 registry 存储配置

对于不提供默认存储的平台，Image Registry Operator 最初将不可用。安装后，您必须配置 registry 使用的存储，这样 Registry Operator 才可用。

示配置生产集群所需的持久性卷的说明。如果适用，显示有关将空目录配置为存储位置的说明，该位置只可用于非生产集群。

另外还提供了在升级过程中使用 **Recreate** rollout 策略来允许镜像 registry 使用块存储类型的说明。

5.2.17.3. 配置块 registry 存储

要允许镜像 registry 在作为集群管理员升级过程中使用块存储类型，您可以使用 **Recreate** rollout 策略。

重要

支持块存储卷，但不建议将其与生产环境中的镜像 registry 一起使用。在块存储上配置 registry 的安装不具有高可用性，因为 registry 无法拥有多个副本。

流程

1. 要将镜像 registry 存储设置为块存储类型，对 registry 进行补丁，使其使用 **Recreate** rollout 策略，并只使用一个（**1**）副本运行：

```
$ oc patch config.imageregistry.operator.openshift.io/cluster --type=merge -p '{"spec":
{"rolloutStrategy":"Recreate","replicas":1}}'
```

2. 为块存储设备置备 PV，并为该卷创建 PVC。请求的块卷使用 ReadWriteOnce（RWO）访问模式。

3. 编辑 registry 配置，使其引用正确的 PVC。

5.2.18. 在用户置备的基础架构上完成安装

完成 Operator 配置后，可以在您提供的基础架构上完成集群安装。

先决条件

- 您的 control plane 已初始化。
- 已完成初始 Operator 配置。

流程

1. 使用以下命令确认所有集群组件都已在线：

```
$ watch -n5 oc get clusteroperators
```

输出示例

```
NAME                                       VERSION AVAILABLE   PROGRESSING   DEGRADED
SINCE
authentication                             4.6.0   True        False         False     3h56m
cloud-credential                           4.6.0   True        False         False     29h
cluster-autoscaler                         4.6.0   True        False         False     29h
config-operator                            4.6.0   True        False         False     6h39m
console                                    4.6.0   True        False         False     3h59m
csi-snapshot-controller                    4.6.0   True        False         False     4h12m
dns                                        4.6.0   True        False         False     4h15m
etcd                                       4.6.0   True        False         False     29h
image-registry                             4.6.0   True        False         False     3h59m
ingress                                    4.6.0   True        False         False     4h30m
insights                                   4.6.0   True        False         False     29h
kube-apiserver                             4.6.0   True        False         False     29h
kube-controller-manager                    4.6.0   True        False         False     29h
kube-scheduler                             4.6.0   True        False         False     29h
kube-storage-version-migrator              4.6.0   True        False         False     4h2m
machine-api                                4.6.0   True        False         False     29h
machine-approver                           4.6.0   True        False         False     6h34m
machine-config                             4.6.0   True        False         False     3h56m
marketplace                                4.6.0   True        False         False     4h2m
monitoring                                 4.6.0   True        False         False     6h31m
network                                    4.6.0   True        False         False     29h
node-tuning                                4.6.0   True        False         False     4h30m
openshift-apiserver                        4.6.0   True        False         False     3h56m
openshift-controller-manager               4.6.0   True        False         False     4h36m
openshift-samples                          4.6.0   True        False         False     4h30m
operator-lifecycle-manager                 4.6.0   True        False         False     29h
operator-lifecycle-manager-catalog         4.6.0   True        False         False     29h
operator-lifecycle-manager-packageserver   4.6.0   True        False         False     3h59m
service-ca                                 4.6.0   True        False         False     29h
storage                                    4.6.0   True        False         False     4h30m
```

或者，通过以下命令，如果所有集群都可用您会接到通知。它还检索并显示凭证：

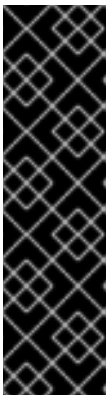
```
$ ./openshift-install --dir <installation_directory> wait-for install-complete 1
```

1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

输出示例

```
INFO Waiting up to 30m0s for the cluster to initialize...
```

Cluster Version Operator 完成从 Kubernetes API 服务器部署 OpenShift Container Platform 集群时，命令运行成功。



重要

- 安装程序生成的 Ignition 配置文件包含在 24 小时后过期的证书，然后在过期时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外情况是，您需要手动批准待处理的 **node-bootstrapper** 证书签名请求（CSR）来恢复 kubelet 证书。如需更多信息，请参阅*从过期的 control plane 证书中恢复*的文档。
- 建议您在生成 12 小时后使用 Ignition 配置文件，因为集群安装后 24 小时证书从 16 小时轮转至 22 小时。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中运行证书更新时避免安装失败。

2. 确认 Kubernetes API 服务器正在与 pod 通信。

   a. 要查看所有 pod 的列表，请使用以下命令：

   ```
   $ oc get pods --all-namespaces
   ```

   输出示例

   ```
   NAMESPACE                       NAME                                        READY   STATUS
   RESTARTS   AGE
   openshift-apiserver-operator    openshift-apiserver-operator-85cb746d55-zqhs8   1/1
   Running    1          9m
   openshift-apiserver             apiserver-67b9g                               1/1     Running     0
   3m
   openshift-apiserver             apiserver-ljcmx                               1/1     Running     0
   1m
   openshift-apiserver             apiserver-z25h4                               1/1     Running     0
   2m
   openshift-authentication-operator authentication-operator-69d5d8bf84-vh2n8        1/1
   Running    0          5m
   ...
   ```

   b. 使用以下命令，查看上一命令的输出中所列 pod 的日志：

   ```
   $ oc logs <pod_name> -n <namespace>
   ```
   (1)

   (1) 指定 pod 名称和命名空间，如上一命令的输出中所示。

   如果 pod 日志显示，Kubernetes API 服务器可以与集群机器通信。

5.2.19. OpenShift Container Platform 的 Telemetry 访问

在 OpenShift Container Platform 4.6 中，默认运行的 Telemetry 服务提供有关集群健康状况和成功更新的指标，需要访问互联网。如果您的集群连接到互联网，Telemetry 会自动运行，而且集群会注册到 OpenShift Cluster Manager。

确认 OpenShift Cluster Manager 清单正确后，可以由 Telemetry 自动维护，也可以使用 OpenShift Cluster Manager 手动维护，使用订阅监控来跟踪帐户或多集群级别的 OpenShift Container Platform 订阅。

其他资源

- 有关 Telemetry 服务的更多信息，请参阅关于远程健康监控。

5.2.20. 后续步骤

- 自定义集群。
- 如果需要，您可以选择不使用远程健康报告。
- 设置 registry 并配置 registry 存储。

5.3. 在受限网络中的裸机上安装集群

在 OpenShift Container Platform 版本 4.6 中，您可以在受限网络中置备的裸机基础架构上安装集群。

重要

虽然您可能能够按照此流程在虚拟化或云环境中部署集群，但您必须清楚非裸机平台的其他注意事项。在尝试在此类环境中安装 OpenShift Container Platform 集群前，请参阅有关在未经测试的平台上部署 OpenShift Container Platform 的指南中的信息。

5.3.1. 先决条件

- 在镜像主机上创建镜像 registry，并获取您的 OpenShift Container Platform 版本的 **imageContentSources** 数据。

  重要

  由于安装介质位于堡垒主机上，因此请使用该计算机完成所有安装步骤。

- 为集群置备持久性存储。若要部署私有镜像 registry，您的存储必须提供 ReadWriteMany 访问模式。
- 查看有关 OpenShift Container Platform 安装和更新流程的详细信息。
- 如果使用防火墙并计划使用遥测（telemetry），您必须将防火墙配置为允许集群需要访问的站点。

  注意

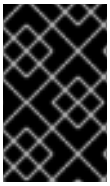
  如果您要配置代理，请务必也要查看此站点列表。

5.3.2. 关于在受限网络中安装

在 OpenShift Container Platform 4.6 中，可以执行不需要有效的互联网连接来获取软件组件的安装。受限网络安装可使用安装程序置备的基础架构或用户置备的基础架构完成，具体取决于您要安装集群的云平台。

如果选择在云平台中执行受限网络安装，仍然需要访问其云 API。有些云功能，比如 Amazon Web Service 的 Route 53 DNS 和 IAM 服务，需要访问互联网。根据您的网络，在裸机硬件或 VMware vSphere 上安装时可能需要较少的互联网访问。

要完成受限网络安装，您必须创建一个 registry，镜像 OpenShift Container Platform registry 的内容并包含其安装介质。您可以在堡垒主机上创建此镜像，该主机可同时访问互联网和您的封闭网络，也可以使用满足您的限制条件的其他方法。



重要

由于用户置备安装配置的复杂性，在尝试使用用户置备的基础架构受限网络安装前，请考虑完成标准用户置备的基础架构安装。通过完成此测试安装，您可以更轻松地隔离和排查您在受限网络中安装时可能出现的问题。

5.3.2.1. 其他限制

受限网络中的集群还有以下额外限制：

- **ClusterVersion** 状态包含一个 **Unable to retrieve available updates** 错误。
- 默认情况下，您无法使用 Developer Catalog 的内容，因为您无法访问所需的镜像流标签。

5.3.3. OpenShift Container Platform 的互联网访问

在 OpenShift Container Platform 4.6 中，您需要访问互联网来获得用来安装集群的镜像。

您必须具有以下互联网访问权限：

- 访问 OpenShift Cluster Manager 以下载安装程序并执行订阅管理。如果集群可以访问互联网，并且没有禁用 Telemetry，该服务会自动授权您的集群。
- 访问 Quay.io，以获取安装集群所需的软件包。
- 获取执行集群更新所需的软件包。

重要

如果您的集群无法直接访问互联网，则可以在置备的某些类基础架构上执行受限网络安装。在此过程中，您要下载所需的内容，并使用它在镜像 registry（mirror registry） 中填充安装集群并生成安装程序所需的软件包。对于某些安装类型，集群要安装到的环境不需要访问互联网。在更新集群之前，要更新 registry 镜像系统中的内容。

5.3.4. 具有用户置备基础架构的集群的机器要求

对于含有用户置备的基础架构的集群，您必须部署所有所需的机器。

5.3.4.1. 所需的机器

最小的 OpenShift Container Platform 集群需要下列主机：

- 一个临时 bootstrap 机器
- 三台 control plane 或 master 机器

- 至少两台计算机器，也称为 worker 机器。如果您正在运行三节点集群，则支持运行零个计算机器。不支持运行一台计算机器。

注意

集群要求 bootstrap 机器在三台 control plane 机器上部署 OpenShift Container Platform 集群。您可在安装集群后删除 bootstrap 机器。

重要

要保持集群的高可用性，请将独立的物理主机用于这些集群机器。

bootstrap 和 control plane 机器必须使用 Red Hat Enterprise Linux CoreOS (RHCOS) 作为操作系统。但是，计算机器可以在 Red Hat Enterprise Linux CoreOS(RHCOS)或 Red Hat Enterprise Linux(RHEL)7.9 之间进行选择。

请注意，RHCOS 基于 Red Hat Enterprise Linux（RHEL） 8，并继承其所有硬件认证和要求。请查看Red Hat Enterprise Linux 技术功能及限制 。

5.3.4.2. 网络连接要求

所有 Red Hat Enterprise Linux CoreOS (RHCOS) 机器在启动过程中需要 **initramfs** 中的网络从 Machine Config Server 获取 Ignition 配置文件。在初次启动过程中，需要一个 DHCP 服务器或设置了静态 IP 地址来建立网络连接，以下载它们的 Ignition 配置文件。另外，集群中的每个 OpenShift Container Platform 节点都必须有权访问网络时间协议（NTP）服务器。如果 DHCP 服务器提供 NTP 服务器信息，Red Hat Enterprise Linux CoreOS（RHCOS）机器上的 chrony 时间服务会读取信息，并可与 NTP 服务器同步时钟。

5.3.4.3. 最低资源要求

每台集群机器都必须满足以下最低要求：

表 5.30. 最低资源要求

机器	操作系统	CPU [1]	RAM	存储	IOPS [2]
bootstrap	RHCOS	4	16 GB	100 GB	300
Control plane	RHCOS	4	16 GB	100 GB	300
Compute	RHCOS 或 RHEL 7.9	2	8 GB	100 GB	300

1. 当未启用并发多线程(SMT)或超线程时，一个 CPU 相当于一个物理内核。启用后，使用以下公式来计算对应的比率：（每个内核的线程数）[1] 插槽 = CPU。

2. OpenShift Container Platform 和 Kubernetes 对磁盘性能非常敏感，建议使用更快的存储速度，特别是 control plane 节点上需要 10 ms p99 fsync 持续时间的 etcd。请注意，在许多云平台上，存储大小和 IOPS 可一起扩展，因此您可能需要过度分配存储卷来获取足够的性能。

5.3.4.4. 证书签名请求管理

在使用您置备的基础架构时，集群只能有限地访问自动机器管理，因此您必须提供一种在安装后批准集群证书签名请求 (CSR) 的机制。**kube-controller-manager** 只能批准 kubelet 客户端 CSR。 **machine-approver** 无法保证使用 kubelet 凭证请求的提供证书的有效性，因为它不能确认是正确的机器发出了该请求。您必须决定并实施一种方法，以验证 kubelet 提供证书请求的有效性并进行批准。

5.3.5. 创建用户置备的基础架构

在部署采用用户置备的基础架构的 OpenShift Container Platform 集群前，您必须创建底层基础架构。

先决条件

- 在为集群创建支持基础架构之前，请参阅OpenShift Container Platform 4.x Tested Integrations页。

流程

1. 在每个节点上配置 DHCP 或设置静态 IP 地址。
2. 提供所需的负载均衡器。
3. 配置机器的端口。
4. 配置 DNS。
5. 确保网络可以正常工作。

5.3.5.1. 用户置备的基础架构对网络的要求

所有 Red Hat Enterprise Linux CoreOS（RHCOS）机器在启动过程中需要 **initramfs** 中的网络从机器配置服务器获取 Ignition 配置。

在初次启动过程中，需要一个 DHCP 服务器或集群中的每个机器都设置了静态 IP 地址来建立网络连接，以下载它们的 Ignition 配置文件。

建议您使用 DHCP 服务器为集群进行长期机器管理。确保 DHCP 服务器已配置为向集群机器提供持久 IP 地址和主机名。

Kubernetes API 服务器必须能够解析集群机器的节点名称。如果 API 服务器和 worker 节点位于不同的区域中，您可以配置默认 DNS 搜索区域，以便 API 服务器能够解析节点名称。另一种支持的方法是始终在节点对象和所有 DNS 请求中使用完全限定域名来指代主机。

您必须配置机器间的网络连接，以便集群组件进行通信。每台机器都必须能够解析集群中所有其他机器的主机名。

表 5.31. 所有机器到所有机器

协议	端口	描述
ICMP	N/A	网络可访问性测试
TCP	**1936**	指标
	9000-**9999**	主机级别的服务，包括端口 **9100**-**9101** 上的节点导出器和端口 **9099** 上的 Cluster Version Operator。

协议	端口	描述
	10250-10259	Kubernetes 保留的默认端口
	10256	openshift-sdn
UDP	**4789**	VXLAN 和 Geneve
	6081	VXLAN 和 Geneve
	9000-9999	主机级别的服务，包括端口 **9100-9101** 上的节点导出器。
TCP/UDP	**30000-32767**	Kubernetes 节点端口

表 5.32. 要通过控制平面的所有机器

协议	端口	描述
TCP	**6443**	Kubernetes API

表 5.33. control plane 机器到 control plane 机器

协议	端口	描述
TCP	**2379-2380**	etcd 服务器和对等端口

网络拓扑要求
您为集群置备的基础架构必须满足下列网络拓扑要求。

负载均衡器
在安装 OpenShift Container Platform 前，您必须置备两个满足以下要求的负载均衡器：

1. **API 负载均衡器**：提供一个通用端点，供用户（包括人和机器）与平台交互和配置。配置以下条件：

   - 只适用于第 4 层负载均衡。这可被称为 Raw TCP、SSL Passthrough 或者 SSL 桥接模式。如果使用 SSL Bridge 模式，必须为 API 路由启用 Server Name Indication（SNI）。
   - 无状态负载平衡算法。这些选项根据负载均衡器的实现而有所不同。

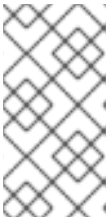
   重要

   不要为 API 负载均衡器配置会话持久性。

   在负载均衡器的前端和后台配置以下端口：

   表 5.34. API 负载均衡器

端口	后端机器（池成员）	内部	外部	描述
6443	Bootstrap 和 control plane.bootstrap 机器初始化集群 control plane 后，您要从负载均衡器中删除 bootstrap 机器。您必须为 API 服务器健康检查探测配置 **/readyz** 端点。	X	X	Kubernetes API 服务器
22623	Bootstrap 和 control plane.bootstrap 机器初始化集群 control plane 后，您要从负载均衡器中删除 bootstrap 机器。	X		机器配置服务器



注意

负载均衡器必须配置为，从 API 服务器关闭 **/readyz** 端点到从池中删除 API 服务器实例时最多需要 30 秒。在 **/readyz** 返回错误或处于健康状态后的时间范围内，端点必须被删除或添加。每 5 秒或 10 秒探测一次，有两个成功请求处于健康状态，三个成为不健康的请求经过测试。

2. **应用程序入口负载均衡器**:提供来自集群外部的应用程序流量流量的 Ingress 点。配置以下条件：

   - 只适用于第 4 层负载均衡。这可被称为 Raw TCP、SSL Passthrough 或者 SSL 桥接模式。如果使用 SSL Bridge 模式，您必须为 Ingress 路由启用Server Name Indication（SNI）。
   - 建议根据可用选项以及平台上托管的应用程序类型，使用基于连接的或者基于会话的持久性。

   在负载均衡器的前端和后台配置以下端口：

表 5.35. 应用程序入口负载均衡器

端口	后端机器（池成员）	内部	外部	描述
443	默认运行入口路由器 Pod、计算或 worker 的机器。	X	X	HTTPS 流量
80	默认运行入口路由器 Pod、计算或 worker 的机器。	X	X	HTTP 流量

提示

如果负载均衡器可以看到客户端的真实 IP 地址，启用基于 IP 的会话持久性可提高使用端到端 TLS 加密的应用程序的性能。

注意

OpenShift Container Platform 集群需要正确配置入口路由器。control plane 初始化后，您必须配置入口路由器。

NTP 配置

OpenShift Container Platform 集群默认配置为使用公共网络时间协议（NTP）服务器。如果要使用本地企业 NTP 服务器，或者集群部署在断开连接的网络中，您可以将集群配置为使用特定的时间服务器。如需更多信息，请参阅*配置 chrony 时间服务*的文档。

如果 DHCP 服务器提供 NTP 服务器信息，Red Hat Enterprise Linux CoreOS（RHCOS）机器上的 chrony 时间服务会读取信息，并可与 NTP 服务器同步时钟。:!restricted:

其他资源

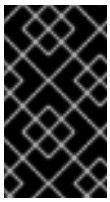
- 配置 chrony 时间服务

5.3.5.2. 用户置备 DNS 要求

DNS 用于名称解析和反向名称解析。DNS A/AAAA 或 CNAME 记录用于名称解析，PTR 记录用于反向解析名称。反向记录很重要，因为 Red Hat Enterprise Linux CoreOS（RHCOS）使用反向记录为所有节点设置主机名。另外，反向记录用于生成 OpenShift Container Platform 需要操作的证书签名请求（CSR）。

采用用户置备的基础架构的 OpenShift Container Platform 集群需要以下 DNS 记录。在每一记录中，**<cluster_name>** 是集群名称，**<base_domain>** 则是您在 **install-config.yaml** 文件中指定的集群基域。完整的 DNS 记录采用如下格式: **<component>.<cluster_name>.<base_domain>.**。

表 5.36. 所需的 DNS 记录

组件	记录	描述
Kubernetes API	**api.<cluster_name>. <base_domain>.**	添加 DNS A/AAAA 或 CNAME 记录，以及 DNS PTR 记录，以识别 control plane 机器的负载均衡器。这些记录必须由集群外的客户端以及集群中的所有节点解析。
	api-int.<cluster_name>. <base_domain>.	添加 DNS A/AAAA 或 CNAME 记录，以及 DNS PTR 记录，以识别 control plane 机器的负载均衡器。这些记录必须可以从集群中的所有节点解析。  **重要** API 服务器必须能够根据在 Kubernetes 中记录的主机名解析 worker 节点。如果 API 服务器无法解析节点名称，则代理的 API 调用会失败，且您无法从 pod 检索日志。
Routes	***.apps.<cluster_name>. <base_domain>.**	添加通配符 DNS A/AAAA 或 CNAME 记录，指向以运行入口路由器 Pod 的机器（默认为 worker 节点）为目标的负载均衡器。这些记录必须由集群外的客户端以及集群中的所有节点解析。
bootstrap	**bootstrap.<cluster_name>. <base_domain>.**	添加 DNS A/AAAA 或 CNAME 记录，以及 DNS PTR 记录来识别 bootstrap 机器。这些记录必须由集群中的节点解析。
Master 主机	**<master><n>. <cluster_name>. <base_domain>.**	DNS A/AAAA 或 CNAME 记录，以识别 control plane 节点（也称为 master 节点）的每台机器。这些记录必须由集群中的节点解析。

组件	记录	描述
Worker 主机	**<worker><n>. <cluster_name>. <base_domain>.**	添加 DNS A/AAAA 或 CNAME 记录，以识别 worker 节点的每台机器。这些记录必须由集群中的节点解析。

提示

您可以使用 **nslookup <hostname>** 命令来验证名称解析。您可以使用 **dig -x <ip_address>** 命令来验证 PTR 记录的反向名称解析。

下面的 BIND 区文件的例子展示了关于名字解析的 A 记录的例子。这个示例的目的是显示所需的记录。这个示例不是为选择一个名称解析服务提供建议。

例 5.5. DNS 区数据库示例

```
$TTL 1W
@ IN SOA ns1.example.com. root (
   2019070700 ; serial
   3H  ; refresh (3 hours)
   30M  ; retry (30 minutes)
   2W  ; expiry (2 weeks)
   1W )  ; minimum (1 week)
 IN NS ns1.example.com.
 IN MX 10 smtp.example.com.
;
;
ns1 IN A 192.168.1.5
smtp IN A 192.168.1.5
;
helper IN A 192.168.1.5
helper.ocp4 IN A 192.168.1.5
;
; The api identifies the IP of your load balancer.
api.ocp4  IN A 192.168.1.5
api-int.ocp4  IN A 192.168.1.5
;
; The wildcard also identifies the load balancer.
*.apps.ocp4  IN A 192.168.1.5
;
; Create an entry for the bootstrap host.
bootstrap.ocp4 IN A 192.168.1.96
;
; Create entries for the master hosts.
master0.ocp4  IN A 192.168.1.97
master1.ocp4  IN A 192.168.1.98
master2.ocp4  IN A 192.168.1.99
;
; Create entries for the worker hosts.
worker0.ocp4  IN A 192.168.1.11
```

```
worker1.ocp4  IN A 192.168.1.7
;
;EOF
```

下面的 BIND 区文件示例显示了反向名字解析的 PTR 记录示例。

例 5.6. 反向记录的 DNS 区数据库示例

```
$TTL 1W
@ IN SOA ns1.example.com. root (
   2019070700 ; serial
   3H  ; refresh (3 hours)
   30M  ; retry (30 minutes)
   2W  ; expiry (2 weeks)
   1W )  ; minimum (1 week)
 IN NS ns1.example.com.
;
; The syntax is "last octet" and the host must have an FQDN
; with a trailing dot.
97 IN PTR master0.ocp4.example.com.
98 IN PTR master1.ocp4.example.com.
99 IN PTR master2.ocp4.example.com.
;
96 IN PTR bootstrap.ocp4.example.com.
;
5 IN PTR api.ocp4.example.com.
5 IN PTR api-int.ocp4.example.com.
;
11 IN PTR worker0.ocp4.example.com.
7 IN PTR worker1.ocp4.example.com.
;
;EOF
```

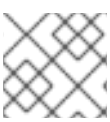
5.3.6. 生成 SSH 私钥并将其添加到代理中

如果要在集群上执行安装调试或灾难恢复，则必须为 **ssh-agent** 和安装程序提供 SSH 密钥。您可以使用此密钥访问公共集群中的 bootstrap 机器来排除安装问题。

注意

在生产环境中，您需要进行灾难恢复和调试。

您可以使用此密钥以 **core** 用户身份通过 SSH 连接到 master 节点。在部署集群时，此密钥会添加到 **core** 用户的 **~/.ssh/authorized_keys** 列表中。



注意

您必须使用一个本地密钥，而不要使用在特定平台上配置的密钥，如 AWS 密钥对。

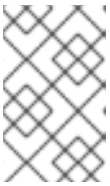
流程

1. 如果还没有为计算机上免密码身份验证而配置的 SSH 密钥，请创建一个。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ ssh-keygen -t ed25519 -N '' \
  -f <path>/<file_name> 1
```

1 指定新 SSH 密钥的路径和文件名，如 **~/.ssh/id_rsa**。如果您已有密钥对，请确保您的公钥位于 **~/.ssh** 目录中。

运行此命令会在指定的位置生成不需要密码的 SSH 密钥。



注意

如果您计划在 **x86_64** 架构中安装使用 FIPS 验证的/Modules in Process 加密库的 OpenShift Container Platform 集群，不要创建使用 **ed25519** 算法的密钥。反之，创建一个使用 **rsa** 或 **ecdsa** 算法的密钥。

2. 作为后台任务启动 **ssh-agent** 进程：

```
$ eval "$(ssh-agent -s)"
```

输出示例

```
Agent pid 31874
```

注意

如果您的集群采用 FIPS 模式，则只使用 FIPS 兼容算法来生成 SSH 密钥。密钥必须是 RSA 或 ECDSA。

3. 将 SSH 私钥添加到 **ssh-agent**：

```
$ ssh-add <path>/<file_name> 1
```

输出示例

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

1 指定 SSH 私钥的路径和文件名，如 **~/.ssh/id_rsa**

后续步骤

- 在安装 OpenShift Container Platform 时，为安装程序提供 SSH 公钥。如果在您置备的基础架构上安装集群，您必须将此密钥提供给集群的机器。

5.3.7. 手动创建安装配置文件

对于使用用户置备的基础架构的 OpenShift Container Platform 安装，您必须手动生成安装配置文件。

先决条件

- 获取 OpenShift Container Platform 安装程序和集群的访问令牌。
- 获取命令输出中的 **imageContentSources** 部分来镜像存储库。
- 获取您的镜像 registry 的证书内容。

流程

1. 创建用来存储您所需的安装资产的安装目录：

```
$ mkdir <installation_directory>
```

重要

您必须创建目录。一些安装信息，如 bootstrap X.509 证书，有较短的过期间隔，因此不要重复使用安装目录。如果要重复使用另一个集群安装中的个别文件，可以将其复制到您的目录中。但是，一些安装数据的文件名可能会在发行版本之间有所改变。从 OpenShift Container Platform 老版本中复制安装文件时要格外小心。

2. 自定义以下 **install-config.yaml** 文件模板，并将它保存到 **<installation_directory>** 中。

注意

此配置文件必须命名为 **install-config.yaml**。

- 除非使用 RHCOS 默认信任的 registry，如 **docker.io**，否则必须在 **additionalTrustBundle** 部分中提供镜像存储库的证书内容。在大多数情况下，必须为您的镜像提供证书。
- 您必须包含命令输出中的 **imageContentSources** 部分，才能镜像存储库。

3. 备份 **install-config.yaml** 文件，以便用于安装多个集群。

重要

install-config.yaml 文件会在安装过程的下一步骤中消耗掉。现在必须备份它。

5.3.7.1. 安装配置参数

在部署 OpenShift Container Platform 集群前，您可以提供参数值，以描述托管集群的云平台的帐户并选择性地自定义集群平台。在创建 **install-config.yaml** 安装配置文件时，您可以通过命令行来提供所需的参数的值。如果要自定义集群，可以修改 **install-config.yaml** 文件来提供关于平台的更多信息。

注意

安装之后，您无法修改 **install-config.yaml** 文件中的这些参数。

重要

openshift-install 命令不验证参数的字段名称。如果指定了不正确的名称，则不会创建相关的文件或对象，且不会报告错误。确保所有指定的参数的字段名称都正确。

5.3.7.1.1. 所需的配置参数

下表描述了所需的安装配置参数：

表 5.37. 所需的参数

参数	描述	值
apiVersion	**install-config.yaml** 内容的 API 版本。当前版本是 **v1**。安装程序还可能支持旧的 API 版本。	字符串
baseDomain	云供应商的基域。此基础域用于创建到 OpenShift Container Platform 集群组件的路由。集群的完整 DNS 名称是 **baseDomain** 和 **metadata.name** 参数值的组合，其格式为 **<metadata.name>.<baseDomain>**。	完全限定域名或子域名，如 **example.com**。
metadata	Kubernetes 资源 **ObjectMeta**，其中只消耗 **name** 参数。	对象
metadata.name	集群的名称。集群的 DNS 记录是 **{{.metadata.name}}.{{.baseDomain}}** 的子域。	小写字母,连字符(-)和句点(.)的字符串，如 **dev**。
platform	执行安装的具体平台配置：**aws**、**baremetal**、**azure**、**openstack**、**ovirt**、**vsphere**。有关 **platform.<platform>** 参数的额外信息，请参考下表来了解您的具体平台。	对象

pullSecret	从 Red Hat OpenShift Cluster Manager 获取 pull secret，验证从 Quay.io 等服务中下载 OpenShift Container Platform 组件的容器镜像。	{
		"auths":{
		"cloud.openshift.com":{
		"auth":"b3Blb=",
		"email":"you@example.com"
		},
		"quay.io":{
		"auth":"b3Blb=",
		"email":"you@example.com"
		}
		}
		}

5.3.7.1.2. 网络配置参数

您可以根据现有网络基础架构的要求自定义安装配置。例如，您可以扩展集群网络的 IP 地址块，或者提供不同于默认值的不同 IP 地址块。

只支持 IPv4 地址。

表 5.38. 网络参数

参数	描述	值
networking	集群网络的配置。	对象  **注意** 您不能在安装后修改 **networking** 对象指定的参数。
networking.networkType	要安装的集群网络供应商 Container Network Interface（CNI）插件。	**OpenShiftSDN** 或 **OVNKubernetes**。默认值为 **OpenShiftSDN**。
networking.clusterNetwork	pod 的 IP 地址块。 默认值为 **10.128.0.0/14**，主机前缀为 **/23**。 如果您指定多个 IP 地址块，则块不得互相重叠。	一个对象数组。例如： networking: clusterNetwork: - cidr: 10.128.0.0/14 hostPrefix: 23
networking.clusterNetwork.cidr	使用 **networking.clusterNetwork** 时需要此项。IP 地址块。 一个 IPv4 网络。	使用 CIDR 形式的 IP 地址块。IPv4 块的前缀长度介于 **0** 到 **32** 之间。
networking.clusterNetwork.hostPrefix	分配给每个单独节点的子网前缀长度。例如，如果 **hostPrefix** 设为 **23**，则每个节点从所给的 **cidr** 中分配一个 **/23** 子网。**hostPrefix** 值 **23** 提供 510（2^(32 - 23)- 2）个 pod IP 地址。	子网前缀。 默认值为 **23**。
networking.serviceNetwork	服务的 IP 地址块。默认值为 **172.30.0.0/16**。 OpenShift SDN 和 OVN-Kubernetes 网络供应商只支持服务网络的一个 IP 地址块。	CIDR 格式具有 IP 地址块的数组。例如： networking: serviceNetwork: - 172.30.0.0/16

参数	描述	值
networking.machineNetwork	机器的 IP 地址块。 如果您指定多个 IP 地址块，则块不得互相重叠。	一个对象数组。例如： networking: machineNetwork: - cidr: 10.0.0.0/16
networking.machineNetwork.cidr	使用 **networking.machineNetwork** 时需要。IP 地址块。libvirt 以外的所有平台的默认值为 **10.0.0.0/16**。对于 libvirt，默认值为 **192.168.126.0/24**。	CIDR 表示法中的 IP 网络块。 例如： **10.0.0.0/16**。 **注意** 将 **networking.machineNetwork** 设置为与首选 NIC 所在的 CIDR 匹配。

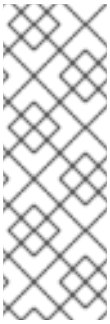
5.3.7.1.3. 可选配置参数

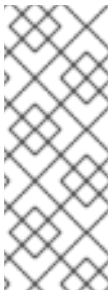
下表描述了可选安装配置参数：

表 5.39. 可选参数

参数	描述	值
additionalTrustBundle	添加到节点可信证书存储中的 PEM 编码 X.509 证书捆绑包。配置了代理时，也可以使用这个信任捆绑包。	字符串
compute	组成计算节点的机器的配置。	machine-pool 对象的数组。详情请查看以下"Machine-pool"表。
compute.architecture	决定池中机器的指令集合架构。目前不支持异构集群，因此所有池都必须指定相同的架构。有效值为 **amd64** （默认值）。	字符串
compute.hyperthreading	是否在计算机器上启用或禁用并发多线程或**超线程**。默认情况下，启用并发多线程以提高机器内核的性能。 **重要** 如果禁用并发多线程，请确保在容量规划时考虑到机器性能可能会显著降低的问题。	**Enabled** 或 **Disabled**

参数	描述	值
compute.name	使用 **compute** 时需要此值。机器池的名称。	**worker**
compute.platform	使用 **compute** 时需要此值。使用此参数指定托管 worker 机器的云供应商。此参数值必须与 **controlPlane.platform** 参数值匹配。	**aws**、**azure**、**gcp**、**openstack**、**ovirt**、**vsphere** 或 **{}**
compute.replicas	要置备的计算机器数量，也称为 worker 机器。	大于或等于 **2** 的正整数。默认值为 **3**。
controlPlane	组成 control plane 的机器的配置。	**MachinePool** 对象的数组。详情请查看以下"Machine-pool"表。
controlPlane.architecture	决定池中机器的指令集合架构。目前不支持异构集群，因此所有池都必须指定相同的架构。有效值为 **amd64** （默认值）。	字符串
controlPlane.hyperthreading	是否在 control plane 机器上启用或禁用并发多线程或**超线程**。默认情况下，启用并发多线程以提高机器内核的性能。  **重要** 如果禁用并发多线程，请确保在容量规划时考虑到机器性能可能会显著降低的问题。	**Enabled** 或 **Disabled**
controlPlane.name	使用 **controlPlane** 时需要。机器池的名称。	**master**
controlPlane.platform	使用 **controlPlane** 时需要。使用此参数指定托管 control plane 机器的云供应商。此参数值必须与 **compute.platform** 参数值匹配。	**aws**、**azure**、**gcp**、**openstack**、**ovirt**、**vsphere** 或 **{}**
controlPlane.replicas	要置备的 control plane 机器数量。	唯一支持的值是 **3**，它是默认值。

参数	描述	值
credentialsMode	Cloud Credential Operator（CCO）模式。如果没有指定任何模式，CCO 会动态地尝试决定提供的凭证的功能，在支持多个模式的平台上使用 mint 模式。  **注意** 不是所有 CCO 模式都支持所有云供应商。如需有关 CCO 模式的更多信息，请参阅*Red Hat Operator 参考指南*内容中的 *Cloud Credential Operator* 条目。	**Mint**、**Passthrough**、**Manual** 或空字符串(**""**)。
fips	启用或禁用 FIPS 模式。默认为 **false**（禁用）。如果启用了 FIPS 模式，运行 OpenShift Container Platform 的 Red Hat Enterprise Linux CoreOS (RHCOS) 机器会绕过默认的 Kubernetes 加密套件，并使用由 RHCOS 提供的加密模块。  **重要** 只有在 **x86_64** 架构中的 OpenShift Container Platform 部署支持 FIPS 验证的/Modules in Process 加密库。  **注意** 如果使用 Azure File 存储，则无法启用 FIPS 模式。	**false** 或 **true**
imageContentSources	release-image 内容的源和仓库。	对象数组。包括一个 **source** 以及可选的 **mirrors**，如下表所示。
imageContentSources.source	使用 **imageContentSources** 时需要。指定用户在镜像拉取规格中引用的仓库。	字符串

参数	描述	值
imageContentSources.mirrors	指定可能还包含同一镜像的一个或多个仓库。	字符串数组
publish	如何发布或公开集群的面向用户的端点，如 Kubernetes API、OpenShift 路由。	**Internal** 或 **External**。默认值为 **External**。 在非云平台上不支持将此字段设置为 **Internal**。  **重要** 如果将字段的值设为 **Internal**，集群将无法运行。如需更多信息，请参阅 BZ#1953035。
sshKey	用于验证集群机器访问的 SSH 密钥或密钥。  **注意** 对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 **ssh-agent** 进程使用的 SSH 密钥。	一个或多个密钥。例如： sshKey: <key1> <key2> <key3>

5.3.7.2. 裸机 install-config.yaml 文件示例

您可以自定义 **install-config.yaml** 文件，以指定有关 OpenShift Container Platform 集群平台的更多信息，或修改所需参数的值。

```
apiVersion: v1
baseDomain: example.com (1)
compute: (2)
- hyperthreading: Enabled (3)
  name: worker
  replicas: 0 (4)
controlPlane: (5)
  hyperthreading: Enabled (6)
  name: master
  replicas: 3 (7)
metadata:
  name: test (8)
networking:
```

```
  clusterNetwork:
  - cidr: 10.128.0.0/14 9
    hostPrefix: 23 10
  networkType: OpenShiftSDN
  serviceNetwork: 11
  - 172.30.0.0/16
platform:
  none: {} 12
fips: false 13
pullSecret: '{"auths":{"<local_registry>": {"auth": "<credentials>","email": "you@example.com"}}}' 14
sshKey: 'ssh-ed25519 AAAA...' 15
additionalTrustBundle: | 16
  -----BEGIN CERTIFICATE-----
  ZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZ
  -----END CERTIFICATE-----
imageContentSources: 17
- mirrors:
  - <local_registry>/<local_repository_name>/release
  source: quay.io/openshift-release-dev/ocp-release
- mirrors:
  - <local_registry>/<local_repository_name>/release
  source: quay.io/openshift-release-dev/ocp-v4.0-art-dev
```

1 集群的基域。所有 DNS 记录都必须是这个基域的子域，并包含集群名称。

2 5 **controlPlane** 部分是一个单映射，但 **compute** 部分是一系列映射。为满足不同数据结构的要求，**compute** 部分的第一行必须以连字符 **-** 开头，**controlPlane** 部分的第一行则不可以连字符开头。只使用一个 control plane 池。

3 6 是否要启用或禁用并发多线程（SMT）或**超线程**。默认情况下，启用 SMT 可提高机器内核的性能。您可以通过将参数值设为 **Disabled** 来禁用。如果禁用 SMT，则必须在所有集群机器中禁用它，其中包括 control plane 和计算机器。

注意

默认启用并发多线程（SMT）。如果在 BIOS 设置中没有启用 SMT，**hyperthreading** 参数不会起作用。

重要

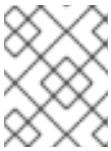
如果您禁用 **hyperthreading**（无论是在 BIOS 中还是在 **install-config.yaml** 中），请确保您对可能会造成的机器性能显著降低的情况有所考虑。

4 **replicas** 参数的值必须设置为 **0**。此参数控制集群为您创建和管理的 worker 数量，使用用户置备的基础架构时集群不会执行这些功能。在完成 OpenShift Container Platform 安装前，您必须手动为集群部署 worker 机器。

7 您添加到集群的 control plane 机器数量。由于集群将这个值用作集群中 etcd 端点的数量，因此该值必须与您部署的 control plane 机器数量匹配。

8 您在 DNS 记录中指定的集群名称。

9 从中分配 pod IP 地址的 IP 地址块。此块不得与现有的物理网络重叠。这些 IP 地址用于 pod 网络。如果您需要从外部网络访问 pod，请配置负载均衡器和路由器来管理流量。



注意

类 E CIDR 范围保留给以后使用。要使用 Class E CIDR 范围，您必须确保您的网络环境接受 Class E CIDR 范围内的 IP 地址。

10 分配给每个单独节点的子网前缀长度。例如，如果 **hostPrefix** 设为 **23**，则每个节点从所给的 **cidr** 中分配一个 **/23** 子网，这样就能有 510 (2^(32 - 23) - 2) 个 Pod IP 地址。如果您需要从外部网络访问节点，请配置负载均衡器和路由器来管理流量。

11 用于服务 IP 地址的 IP 地址池。您只能输入一个 IP 地址池。此块不得与现有的物理网络重叠。如果您需要从外部网络访问服务，请配置负载均衡器和路由器来管理流量。

12 您必须将平台设置为 **none**。您不能为您的平台提供额外的平台配置变量。

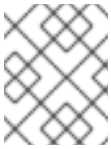
13 是否启用或禁用 FIPS 模式。默认情况下不启用 FIPS 模式。如果启用了 FIPS 模式，运行 OpenShift Container Platform 的 Red Hat Enterprise Linux CoreOS (RHCOS) 机器会绕过默认的 Kubernetes 加密套件，并使用由 RHCOS 提供的加密模块。

重要

只有在 **x86_64** 架构中的 OpenShift Container Platform 部署支持 FIPS 验证的/Modules in Process 加密库。

14 对于 **<local_registry>**，请指定 registry 域名，以及您的镜像 registry 用来提供内容的可选端口。例如： **registry.example.com** 或者 **registry.example.com:5000**。使用 **<credentials>** 为您生成的镜像 registry 指定 base64 编码的用户名和密码。

15 Red Hat Enterprise Linux CoreOS (RHCOS) 中 **core** 用户的默认 SSH 密钥的公钥部分。



注意

对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 **ssh-agent** 进程使用的 SSH 密钥。

16 提供用于镜像 registry 的证书文件内容。

17 提供命令输出中的 **imageContentSources** 部分来镜像存储库。

5.3.7.3. 在安装过程中配置集群范围代理

生产环境可能会拒绝直接访问互联网，而是提供 HTTP 或 HTTPS 代理。您可以通过在 **install-config.yaml** 文件中配置代理设置，将新的 OpenShift Container Platform 集群配置为使用代理。

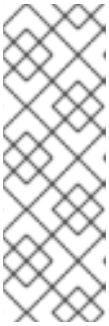
注意

对于裸机安装，如果您没有从 **install-config.yaml** 文件中的 **networking.machineNetwork[].cidr** 字段指定的范围分配节点 IP 地址，您必须将其包括在 **proxy.noProxy** 字段中。

先决条件

- 您有一个现有的 **install-config.yaml** 文件。

- 您检查了集群需要访问的站点，并决定是否需要绕过代理。默认情况下代理所有集群出口流量，包括对托管云供应商 API 的调用。您需要将站点添加到 **Proxy** 对象的 **spec.noProxy** 字段来绕过代理。



注意

Proxy 对象 **status.noProxy** 字段使用安装配置中的 **networking.machineNetwork[].cidr**、**networking.clusterNetwork[].cidr** 和 **networking.serviceNetwork[]** 字段的值填充。

对于在 Amazon Web Services(AWS)、Google Cloud Platform(GCP)、Microsoft Azure 和 Red Hat OpenStack Platform(RHOSP)上安装, **Proxy** 对象 **status.noProxy** 字段也会使用实例元数据端点填充(**169.254.169.254**)。

流程

1. 编辑 **install-config.yaml** 文件并添加代理设置。例如：

```
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: example.com 3
additionalTrustBundle: | 4
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
...
```

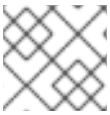
1. 用于创建集群外 HTTP 连接的代理 URL。URL 必须是 **http**。
2. 用于创建集群外 HTTPS 连接的代理 URL。
3. 要排除在代理中的目标域名、IP 地址或其他网络 CIDR 的逗号分隔列表。在域前面加 **.** 来仅匹配子域。例如： **.y.com** 匹配 **x.y.com**，但不匹配 **y.com**。使用 ***** 绕过所有目的地的代理。
4. 如果提供，安装程序会在 **openshift-config 命名空间中生成名为 user-ca- bundle** 的配置映射来保存额外的 CA 证书。如果您提供 **additionalTrustBundle** 和至少一个代理设置，则 **Proxy** 对象会被配置为引用 **trustedCA** 字段中的 **user-ca-bundle** 配置映射。然后，Cluster Network Operator 会创建一个 **trusted-ca-bundle** 配置映射，该配置映射将为 **trustedCA** 参数指定的内容与 RHCOS 信任捆绑包合并。 **additionalTrustBundle** 字段是必需的，除非代理的身份证书由来自 RHCOS 信任捆绑包的颁发机构签名。

注意

安装程序不支持代理的 **readinessEndpoints** 字段。

2. 保存该文件，并在安装 OpenShift Container Platform 时引用。

安装程序会创建一个名为 **cluster** 的集群范围代理，该代理使用提供的 **install-config.yaml** 文件中的代理设置。如果没有提供代理设置，仍然会创建一个 **cluster Proxy** 对象，但它会有一个空 **spec**。



注意

只支持名为 **cluster** 的 **Proxy** 对象，且无法创建额外的代理。

5.3.8. 配置三节点集群

您可在没有 worker 的 OpenShift Container Platform 中安装和运行三节点集群。这为集群管理员和开发人员提供了较小的、效率更高的集群，用于开发、生产及测试。

流程

- 编辑 **install-config.yaml** 文件，将计算副本（也称为 worker 副本）数设为 **0**，如以下 **compute** 小节中所示：

```
compute:
- name: worker
  platform: {}
  replicas: 0
```

5.3.9. 创建 Kubernetes 清单和 Ignition 配置文件

由于您必须修改一些集群定义文件并要手动启动集群机器，因此您必须生成 Kubernetes 清单和 Ignition 配置文件，集群需要这两项来创建其机器。

安装配置文件转换为 Kubernetes 清单。清单嵌套到 Ignition 配置文件中，稍后用于创建集群。

重要

- 安装程序生成的 Ignition 配置文件包含在 24 小时后过期的证书，然后在过期时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外情况是，您需要手动批准待处理的 **node-bootstrapper** 证书签名请求（CSR）来恢复 kubelet 证书。如需更多信息，请参阅*从过期的 control plane 证书中恢复*的文档。
- 建议您在生成 12 小时后使用 Ignition 配置文件，因为集群安装后 24 小时证书从 16 小时轮转至 22 小时。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中运行证书更新时避免安装失败。

先决条件

- 已获得 OpenShift Container Platform 安装程序。对于受限网络安装，这些文件位于您的堡垒主机上。
- 已创建 **install-config.yaml** 安装配置文件。

流程

1. 切换到包含安装程序的目录，并为集群生成 Kubernetes 清单：

```
$ ./openshift-install create manifests --dir <installation_directory> 1
```

1 对于 **<installation_directory>**，请指定含有您创建的 **install-config.yaml** 文件的安装目录。

警告

如果要安装一个三节点集群，请跳过以下步骤，以便 control plane 节点可以调度。

+

重要

当您将 control plane 节点从默认的不可调度配置为可以调度时，需要额外的订阅。这是因为 control plane 节点随后变为 worker 节点。

1. 检查 **<installation_directory>/manifests/cluster-scheduler-02-config.yml** Kubernetes 清单文件中的 **mastersSchedulable** 参数是否已设置为 **false**。此设置可防止在 control plane 机器上调度 pod:
   a. 打开 **<installation_directory>/manifests/cluster-scheduler-02-config.yml** 文件。
   b. 找到 **mastersSchedulable** 参数并确保它被设置为 **false**。
   c. 保存并退出文件。
2. 要创建 Ignition 配置文件，从包含安装程序的目录运行以下命令：

```
$ ./openshift-install create ignition-configs --dir <installation_directory> 1
```

1 对于 **<installation_directory>**，请指定相同的安装目录。

该目录中将生成以下文件：

```
.
├── auth
│   ├── kubeadmin-password
│   └── kubeconfig
├── bootstrap.ign
├── master.ign
├── metadata.json
└── worker.ign
```

5.3.10. 配置 chrony 时间服务

您需要修改 **chrony.conf** 文件的内容来设置 chrony 时间服务（**chronyd**）使用的时间服务器和相关设置，并通过一个机器配置将这些内容传递给节点。

流程

1. 创建 **chrony.conf** 文件的内容并对其进行 base64 编码。例如：

```
$ cat << EOF | base64
```

```
    pool 0.rhel.pool.ntp.org iburst 1
    driftfile /var/lib/chrony/drift
    makestep 1.0 3
    rtcsync
    logdir /var/log/chrony
EOF
```

1 指定任何有效的、可访问的时间源，如 DHCP 服务器提供的时间源。

输出示例

```
ICAgIHNlcnZlciBjbG9jay5yZWRoYXQuY29tIGlidXJzdAogICAgZHJpZnRmaWxlIC92YXIvbGli
L2Nocm9ueS9kcmlmdAogICAgbWFrZXN0ZXAgMS4wIDMKICAgIHJ0Y3N5bmMKICAgIGxvZ2
RpciAv
dmFyL2xvZy9jaHJvbnkK
```

2. 创建 **MachineConfig** 对象文件，将 base64 字符串替换为您刚刚创建的字符串。本例将文件添加到 **master** 节点。您可以将其更改为 **worker**，或为 **worker** 角色创建额外的 MachineConfig。为集群使用的每种机器创建 MachineConfig 文件：

```
$ cat << EOF > ./99-masters-chrony-configuration.yaml
apiVersion: machineconfiguration.openshift.io/v1
kind: MachineConfig
metadata:
  labels:
    machineconfiguration.openshift.io/role: master
  name: 99-masters-chrony-configuration
spec:
  config:
    ignition:
      config: {}
      security:
        tls: {}
      timeouts: {}
      version: 3.1.0
    networkd: {}
    passwd: {}
    storage:
      files:
      - contents:
          source: data:text/plain;charset=utf-
8;base64,ICAgIHNlcnZlciBjbG9jay5yZWRoYXQuY29tIGlidXJzdAogICAgZHJpZnRmaWxlIC92Y
XIvbGliL2Nocm9ueS9kcmlmdAogICAgbWFrZXN0ZXAgMS4wIDMKICAgIHJ0Y3N5bmMKICAg
IGxvZ2RpciAvdmFyL2xvZy9jaHJvbnkK
        mode: 420 1
        overwrite: true
        path: /etc/chrony.conf
  osImageURL: ""
EOF
```

1 为机器配置文件的 **mode** 字段指定数值模式。在创建文件并应用更改后，**模式** 将转换为十进制值。您可以使用 **oc get mc <mc-name> -o yaml** 命令来检查 YAML 文件。

3. 对配置文件做一个备份副本。

4. 使用两种方式之一应用配置：

   - 如果集群还没有启动，在生成清单文件后，将此文件添加到 **<installation_directory>/openshift** 目录中，然后继续创建集群。

   - 如果集群已在运行，请应用该文件：

     ```
     $ oc apply -f ./99-masters-chrony-configuration.yaml
     ```

5.3.11. 安装 RHCOS 并启动 OpenShift Container Platform bootstrap 过程

要在您置备的裸机基础架构上安装 OpenShift Container Platform，您必须在机器上安装 Red Hat Enterprise Linux CoreOS（RHCOS）。安装 RHCOS 时，您必须为 OpenShift Container Platform 安装程序生成的机器类型提供 Ignition 配置文件。如果您配置了合适的网络、DNS 和负载均衡基础架构，OpenShift Container Platform bootstrap 过程会在 RHCOS 机器重启后自动开始。

要在机器上安装 RHCOS，请按照以下步骤使用 ISO 镜像或网络 PXE 启动。

注意

本安装文档中包括的计算节点部署步骤特定于 RHCOS。如果您选择部署基于 RHEL 的计算节点，您将接管所有操作系统生命周期管理和维护，包括执行系统更新、应用补丁和完成所有其他必要的任务。RHEL 7 计算机器的使用已弃用，计划在以后的 OpenShift Container Platform 4 发行版本中删除。

您可以使用以下方法在 ISO 和 PXE 安装过程中配置 RHCOS:

- 内核参数： 您可以使用内核参数来提供特定于安装的信息。例如，您可以指定上传到 HTTP 服务器的 RHCOS 安装文件的位置，以及您要安装的节点类型的 Ignition 配置文件的位置。对于 PXE 安装，您可以使用 **APPEND** 参数将参数传递给实时安装程序的内核。对于 ISO 安装，您可以中断实时安装引导过程来添加内核参数。在这两种安装情况下，您可以使用特殊的 **coreos.inst.*** 参数来指示实时安装程序，以及标准安装引导参数来打开或关闭标准内核服务。

- Ignition 配置：OpenShift Container Platform Ignition 配置文件（***.ign**）特定于您要安装的节点类型。您可以在 RHCOS 安装过程中传递 bootstrap、control plane 或计算节点 Ignition 配置文件的位置，以便在第一次引导时生效。特殊情况下，您可以创建单独的、有限的 Ignition 配置来传递给 Live 系统。该 Ignition 配置可以执行特定任务，如在安装完成后向置备系统报告成功。这个特殊 Ignition 配置由 **coreos-installer** 使用，用于首次启动安装的系统。不要直接向 live ISO 提供标准 control plane 和计算节点 Ignition 配置。

- **coreos-installer**：您可以将 live ISO 安装程序引导到 shell 提示符，这可让您在首次引导前以多种方式准备持久性系统。特别是，您可以运行 **coreos-installer** 命令来识别包括的工件、使用磁盘分区以及设置联网。在有些情况下，您可以配置 live 系统上的功能并将其复制到安装的系统中。

使用 ISO 安装还是 PXE 安装要根据您的具体情况而定。PXE 安装需要可用的 DHCP 服务并进行更多准备，但可以使安装过程更自动化。ISO 安装是一个更手动的过程，如果您设置的机器较多，则可能不方便。

注意

自 OpenShift Container Platform 4.6 起，RHCOS ISO 和其他安装工件支持在带有 4K 扇区的磁盘上安装。

5.3.11.1. 使用 ISO 镜像创建 Red Hat Enterprise Linux CoreOS (RHCOS) 机器

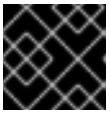
在您置备的基础架构上安装集群前，必须先创建 RHCOS 机器供其使用。您可以使用 ISO 镜像来创建这些机器。

先决条件

- 获取集群的 Ignition 配置文件。
- 具有可从计算机以及您创建的机器访问的 HTTP 服务器的访问权限。

流程

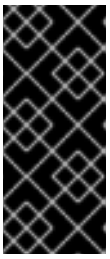
1. 将安装程序创建的 control plane、计算和 bootstrap Ignition 配置文件上传到 HTTP 服务器。记下这些文件的 URL。



重要

如果您计划在安装完成后在集群中添加更多计算机器，请不要删除这些文件。

2. 从 RHCOS 镜像镜像 页面获取您选择的操作系统实例安装方法所需的 RHCOS 镜像。



重要

RHCOS 镜像可能不会随着 OpenShift Container Platform 的每一发行版本都有改变。您必须下载最高版本的镜像，其版本号应小于或等于您安装的 OpenShift Container Platform 版本。如果可用，请使用与 OpenShift Container Platform 版本匹配的镜像版本。此流程只使用 ISO 镜像。此安装类型不支持 RHCOS qcow2 镜像。

ISO 文件名类似以下示例：

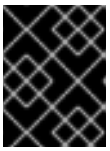
rhcos-<version>-live.<architecture>.iso

3. 使用 ISO 启动 RHCOS 安装。使用如下安装选项之一：
   - 将 ISO 镜像刻录到磁盘并直接启动。
   - 通过 LOM 接口使用 ISO 重定向。
4. 引导 ISO 镜像。您可以中断安装引导过程来添加内核参数。然而，在这个 ISO 过程中，您应该使用 **coreos-installer** 命令而不是添加内核参数。如果您在没有选项或中断的情况下运行 live 安装程序，安装程序将引导至 live 系统上的 shell 提示符，准备好将 RHCOS 安装到磁盘中。
5. 在运行 **coreos-installer** 前，请参阅 *高级 RHCOS 安装参考* 部分，以了解配置功能的不同方法，如网络和磁盘分区。
6. 运行 **coreos-installer** 命令。您至少必须识别节点类型的 Ignition 配置文件位置，以及您要安装到的磁盘位置。下面是一个示例：

```
$ sudo coreos-installer install \
    --ignition-url=https://host/worker.ign /dev/sda
```

7. 安装 RHCOS 后，系统会重启。系统重启过程中，它会应用您指定的 Ignition 配置文件。

8. 继续为集群创建其他机器。



重要

此刻您必须创建 bootstrap 和 control plane 机器。如果 control plane 机器不可调度（这是默认调度），则在安装集群前至少会创建两台计算机器。

5.3.11.2. 通过 PXE 或 iPXE 启动来创建 Red Hat Enterprise Linux CoreOS (RHCOS) 机器

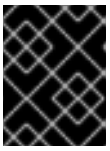
在安装使用手动置备 RHCOS 节点（如裸机）的集群前，您必须创建 RHCOS 机器供其使用。您可以使用 PXE 或 iPXE 启动来创建机器。

先决条件

- 获取集群的 Ignition 配置文件。
- 配置合适的 PXE 或 iPXE 基础架构。
- 具有 HTTP 服务器的访问权限，以便您可从计算机进行访问。

流程

1. 将安装程序创建的 master、worker 和 bootstrap Ignition 配置文件上传到 HTTP 服务器。记下这些文件的 URL。



重要

您可以在 Ignition 配置中添加或更改配置设置，然后将其保存到 HTTP 服务器。如果您计划在安装完成后在集群中添加更多计算机器，请不要删除这些文件。

2. 从 RHCOS 镜像 镜像页面获取 RHCOS **内核、initram fs 和 rootfs** 文件。

重要

RHCOS 工件（artifact）可能不会随着 OpenShift Container Platform 的每个发行版本而改变。您必须下载最高版本的工件，其版本号应小于或等于您安装的 OpenShift Container Platform 版本。这个过程只使用下面描述的正确 **kernel**、**initramfs** 和 **rootfs** 工件。此安装类型不支持 RHCOS qcow2 镜像。

文件名包含 OpenShift Container Platform 版本号。它们类似以下示例：

- **kernel**: **rhcos-<version>-live-kernel-<architecture>**
- **initramfs**: **rhcos-<version>-live-initramfs.<architecture>.img**
- **rootfs**: **rhcos-<version>-live-rootfs.<architecture>.img**

3. 上传引导方法所需的额外文件：
   - 对于传统的 PXE，将 **kernel** 和 **initramfs** 文件上传到 TFTP 服务器，并将 **rootfs** 文件上传到 HTTP 服务器。
   - 对于 iPXE，将 **kernel**、**initram** fs 和 **rootfs** 文件上传到 HTTP 服务器。

重要

如果您计划在安装完成后在集群中添加更多计算机器，请不要删除这些文件。

4. 配置网络启动基础架构，以便在安装 RHCOS 后机器可从本地磁盘启动。

5. 为 RHCOS 镜像配置 PXE 或 iPXE 安装。
针对您的环境修改以下示例菜单条目之一，并验证能否正确访问镜像和 Ignition 文件：

- 对于 PXE：

```
DEFAULT pxeboot
TIMEOUT 20
PROMPT 0
LABEL pxeboot
    KERNEL http://<HTTP_server>/rhcos-<version>-live-kernel-<architecture> (1)
    APPEND initrd=http://<HTTP_server>/rhcos-<version>-live-initramfs.
<architecture>.img coreos.live.rootfs_url=http://<HTTP_server>/rhcos-<version>-live-
rootfs.<architecture>.img coreos.inst.install_dev=/dev/sda
coreos.inst.ignition_url=http://<HTTP_server>/bootstrap.ign (2) (3)
```

(1) 指定上传到 HTTP 服务器的 live **kernel** 文件位置。URL 必须是 HTTP、TFTP 或者 FTP ; 不支持 HTTPS 和 NFS。

(2) 如果您使用多个 NIC，请在 **ip** 选项中指定一个接口。例如，要在名为 **eno1** 的 NIC 上使用 DHCP，请设置 **ip=eno1:dhcp**。

(3) 指定上传到 HTTP 服务器的 RHCOS 文件的位置。**initrd** 参数值是 **initramfs** 文件的位置，**coreos.live.rootfs_url** 参数值是 **rootfs** 文件的位置，**coreos.inst.ignition_url** 参数值是 bootstrap Ignition 配置文件的位置。您还可以在 **APPEND** 行中添加更多内核参数来配置联网或其他引导选项。

注意

这个配置不会在使用图形控制台的机器上启用串口控制台访问。要配置不同的控制台，请在 **APPEND** 行中添加一个或多个 **console=** 参数。例如，添加 **console=tty0 console=ttyS0** 将第一个 PC 串口设置为主控制台，图形控制台作为二级控制台。如需更多信息，请参阅如何在 Red Hat Enterprise Linux 中设置串行终端和（或）控制台？

- 对于 iPXE：

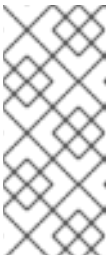
```
kernel http://<HTTP_server>/rhcos-<version>-live-kernel-<architecture> initrd=main
coreos.live.rootfs_url=http://<HTTP_server>/rhcos-<version>-live-rootfs.
<architecture>.img coreos.inst.install_dev=/dev/sda
coreos.inst.ignition_url=http://<HTTP_server>/bootstrap.ign (1) (2)
initrd --name main http://<HTTP_server>/rhcos-<version>-live-initramfs.
<architecture>.img (3)
boot
```

(1) 指定上传到 HTTP 服务器的 RHCOS 文件的位置。**kernel** 参数值是 **kernel** 文件的位置，在 UEFI 系统中引导时需要 **initrd=main** 参数。 **coreos.live.rootfs_url** 参数值是 **rootfs** 文件的位置， **coreos.inst.ignition_url** 参数值则是 bootstrap Ignition 配置文件

的位置。

2 如果您使用多个 NIC，请在 **ip** 选项中指定一个接口。例如，要在名为 **eno1** 的 NIC 上使用 DHCP，请设置 **ip=eno1:dhcp**。

3 指定上传到 HTTP 服务器的 **initramfs** 文件的位置。



注意

这个配置不会在使用图形控制台的机器上启用串口控制台访问。要配置不同的控制台，请在 **kerne** 行中添加一个或多个 **console=** 参数。例如，添加 **console=tty0 console=ttyS0** 将第一个 PC 串口设置为主控制台，图形控制台作为二级控制台。如需更多信息，请参阅如何在 Red Hat Enterprise Linux 中设置串行终端和（或）控制台？

6. 如果使用 PXE UEFI，请执行以下操作：

   a. 提供引导系统所需的 **shim x64.efi 和 grubx64 .efi EFI 二进制文件以及 grub.cfg** 文件。

   - 通过将 RHCOS ISO 挂载到主机，然后将 **images/efiboot.img** 文件挂载到您的主机来提取所需的 EFI 二进制文件：

     ```
     $ mkdir -p /mnt/iso
     ```

     ```
     $ mkdir -p /mnt/efiboot
     ```

     ```
     $ mount -o loop rhcos-installer.x86_64.iso /mnt/iso
     ```

     ```
     $ mount -o loop,ro /mnt/iso/images/efiboot.img /mnt/efiboot
     ```

   - 从 **efiboot.img** 挂载点，将 **EFI/redhat/shimx64.efi** 和 **EFI/redhat/grubx64.efi** 文件复制到 TFTP 服务器中：

     ```
     $ cp /mnt/efiboot/EFI/redhat/shimx64.efi .
     ```

     ```
     $ cp /mnt/efiboot/EFI/redhat/grubx64.efi .
     ```

     ```
     $ umount /mnt/efiboot
     ```

     ```
     $ umount /mnt/iso
     ```

   - 将 RHCOS ISO 中包含的 **EFI/redhat/grub.cfg** 文件复制到您的 TFTP 服务器中。

   b. 编辑 **grub.cfg** 文件使其包含类似如下的参数：

   ```
   menuentry 'Install Red Hat Enterprise Linux CoreOS' --class fedora --class gnu-linux --
   class gnu --class os {
    linuxefi rhcos-<version>-live-kernel-<architecture> coreos.inst.install_dev=/dev/sda
   coreos.live.rootfs_url=http://<HTTP_server>/rhcos-<version>-live-rootfs.
   ```

```
<architecture>.img coreos.inst.ignition_url=http://<HTTP_server>/bootstrap.ign
 initrdefi rhcos-<version>-live-initramfs.<architecture>.img
}
```

其中：

rhcos-<version>-live-kernel-<architecture>

指定上传到 TFTP 服务器的 **内核** 文件。

http://<HTTP_server>/rhcos-<version>-live-rootfs.<architecture>.img

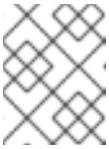
指定上传到 HTTP 服务器的 live rootfs 镜像的位置。

http://<HTTP_server>/bootstrap.ign

指定上传到 HTTP 服务器的 bootstrap Ignition 配置文件的位置。

rhcos-<version>-live-initramfs.<architecture>.img

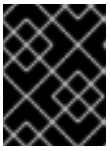
指定上传到 TFTP 服务器的 **initramfs** 文件的位置。



注意

有关如何为 UEFI 引导配置 PXE 服务器的更多信息，请参阅红帽知识库文章：如何为 Red Hat Enterprise Linux 的 UEFI 引导配置/设置 PXE 服务器？

7. 继续为集群创建机器。



重要

此刻您必须创建 bootstrap 和 control plane 机器。如果 control plane 机器不可调度（这是默认调度），则在安装集群前至少会创建两台计算机器。

5.3.11.3. 高级 Red Hat Enterprise Linux CoreOS（RHCOS）安装配置

为 OpenShift Container Platform 手动置备 Red Hat Enterprise Linux CoreOS（RHCOS）节点的一个关键优点是能够进行通过默认的 OpenShift Container Platform 安装方法无法进行的配置。本节介绍了您可以使用的一些技术来进行配置，其中包括：

- 将内核参数传递给实时安装程序
- 从 live 系统手动运行 **coreos-installer**
- 将 Ignition 配置嵌入 ISO 中

本节详述了与 Red Hat Enterprise Linux CoreOS（RHCOS）手动安装的高级配置相关的内容，如磁盘分区、网络以及使用 Ignition 配置的不同方式相关。

5.3.11.3.1. 使用高级网络选项进行 PXE 和 ISO 安装

OpenShift Container Platform 节点的网络默认使用 DHCP 来收集所有必要配置设置。要设置静态 IP 地址或配置特殊的设置，如绑定,您可以执行以下操作之一：

- 引导 live 安装程序时会传递特殊的内核参数。
- 使用机器配置将网络文件复制到安装的系统中。

- 使用 live installer shell 提示配置网络，然后将那些设置复制到安装的系统上，以便在安装的系统第一次引导时生效。

要配置 PXE 或 iPXE 安装，请使用以下选项之一：

- 请参阅"高级 RHCOS 安装参考"表。
- 使用机器配置将网络文件复制到安装的系统中。

要配置 ISO 安装，请使用以下步骤。

流程

1. 引导 ISO 安装程序。
2. 在 live 系统 shell 提示下，使用可用的 RHEL 工具（如 **nmcli** 或 **nmtui**）为 Live 系统配置网络。
3. 运行 **coreos-installer** 命令来安装系统，添加 **--copy-network** 选项来复制网络配置。例如：

```
$ coreos-installer install --copy-network \
    --ignition-url=http://host/worker.ign /dev/sda
```

重要

copy-network 选项只复制 **/etc/NetworkManager/system-connections** 下的网络配置。特别是，它不会复制系统主机名。

4. 重启安装的系统。

5.3.11.3.2. 磁盘分区

磁盘分区是在 Red Hat Enterprise Linux CoreOS（RHCOS）安装过程中在 OpenShift Container Platform 集群节点上创建的。特定架构的每个 RHCOS 节点都使用相同的分区布局，除非默认分区配置被覆盖。在 RHCOS 安装过程中，根文件系统的大小会增大，以使用目标设备中剩余的可用空间。

但是，在安装 OpenShift Container Platform 节点时，在两种情况下您可能需要覆盖默认分区：

- 创建单独的分区：对于在空磁盘中的 greenfield 安装，您可能想要在分区中添加单独的存储。这只在生成 **/var** 或者一个 **/var** 独立分区的子目录（如 **/var/lib/etcd** ）时被正式支持，但不支持两者。

重要

Kubernetes 只支持两个文件系统分区。如果您在原始配置中添加多个分区，Kubernetes 无法监控所有这些分区。

- 保留现有分区：对于 brownfield 安装，您要在现有节点上重新安装 OpenShift Container Platform，并希望保留从之前的操作系统中安装的数据分区，对于 **coreos-installer** 来说，引导选项和选项都允许您保留现有数据分区。

5.3.11.3.2.1. 创建一个独立的 /var 分区

通常情况下，OpenShift Container Platform 的磁盘分区应该留给安装程序。然而，在有些情况下您可能需要在文件系统的一部分中创建独立分区。

OpenShift Container Platform 支持添加单个分区来将存储附加到 **/var** 分区或 **/var** 的子目录。例如：

- **/var/lib/containers**：保存镜像相关的内容，随着更多镜像和容器添加到系统中，它所占用的存储会增加。
- **/var/lib/etcd**：保存您可能希望保持独立的数据，比如 etcd 存储的性能优化。
- **/var**：保存您希望独立保留的数据，用于特定目的（如审计）。

单独存储 **/var** 目录的内容可方便地根据需要对区域扩展存储，并可以在以后重新安装 OpenShift Container Platform 时保持该数据地完整。使用这个方法，您不必再次拉取所有容器，在更新系统时也无法复制大量日志文件。

因为 **/var** 在进行一个全新的 Red Hat Enterprise Linux CoreOS（RHCOS）安装前必需存在，所以这个流程会在 OpenShift Container Platform 安装过程的 **openshift-install** 准备阶段插入的机器配置来设置独立的 **/var** 分区。

流程

1. 创建存放 OpenShift Container Platform 安装文件的目录：

```
$ mkdir $HOME/clusterconfig
```

2. 运行 **openshift-install** 在 **manifest** 和 **openshift** 子目录中创建一组文件。在出现提示时回答系统问题：

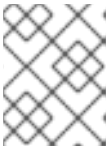
```
$ openshift-install create manifests --dir $HOME/clusterconfig
? SSH Public Key ...
$ ls $HOME/clusterconfig/openshift/
99_kubeadmin-password-secret.yaml
99_openshift-cluster-api_master-machines-0.yaml
99_openshift-cluster-api_master-machines-1.yaml
99_openshift-cluster-api_master-machines-2.yaml
...
```

3. 创建 **MachineConfig** 对象并将其添加到 **openshift** 目录中的一个文件中。例如，把文件命名为 **98-var-partition.yaml**，将磁盘设备名称改为 **worker** 系统中存储设备的名称，并根据情况设置存储大小。这个示例将 **/var** 目录放在一个单独的分区中：

```
apiVersion: machineconfiguration.openshift.io/v1
kind: MachineConfig
metadata:
  labels:
    machineconfiguration.openshift.io/role: worker
  name: 98-var-partition
spec:
  config:
    ignition:
      version: 3.1.0
    storage:
      disks:
      - device: /dev/<device_name> 1
        partitions:
        - label: var
          startMiB: <partition_start_offset> 2
```

```
      sizeMiB: <partition_size> 3
  filesystems:
    - device: /dev/disk/by-partlabel/var
      path: /var
      format: xfs
systemd:
  units:
    - name: var.mount 4
      enabled: true
      contents: |
        [Unit]
        Before=local-fs.target
        [Mount]
        What=/dev/disk/by-partlabel/var
        Where=/var
        Options=defaults,prjquota 5
        [Install]
        WantedBy=local-fs.target
```

1. 要分区的磁盘的存储设备名称。
2. 当在引导磁盘中添加数据分区时，推荐最少使用 25000MB。root 文件系统会自动重新定义大小使其占据所有可用空间（最多到指定的偏移值）。如果没有指定值，或者指定的值小于推荐的最小值，则生成的 root 文件系统会太小，而在以后进行的 RHCOS 重新安装可能会覆盖数据分区的开始部分。
3. 数据分区的大小（以兆字节为单位）。
4. 挂载单元的名称必须与 **Where=** 指令中指定的目录匹配。例如，对于挂载在 **/var/lib/containers** 上的文件系统，该单元必须命名为 **var-lib-containers.mount**。
5. 对于用于容器存储的文件系统，必须启用 **prjquota** 挂载选项。



注意

在创建独立 **/var** 分区时，如果不同的实例类型没有相同的设备名称，则无法将不同的实例类型用于 worker 节点。

4. 再次运行 **openshift-install**，从 **manifest** 和 **openshift** 子目录中的一组文件创建 Ignition 配置：

```
$ openshift-install create ignition-configs --dir $HOME/clusterconfig
$ ls $HOME/clusterconfig/
auth  bootstrap.ign  master.ign  metadata.json  worker.ign
```

现在，可以使用 Ignition 配置文件作为 ISO 或 PXE 手动安装过程的输入来安装 Red Hat Enterprise Linux CoreOS（RHCOS）系统。

5.3.11.3.2.2. 保留现有分区

对于 ISO 安装，您可以在 **coreos-installer** 命令行中添加可让安装程序维护一个或多个现有分区的选项。对于 PXE 安装，您可以 **APPEND coreos.inst.*** 选项来保留分区。

保存的分区可能是来自现有 OpenShift Container Platform 系统中的分区，其中包括了您希望保留的数据分区。以下是几个提示：

- 如果您保存了现有分区，且这些分区没有为 RHCOS 留下足够空间，则安装将失败但不会损害已保存的分区。
- 通过分区标签或数字识别您要保留的磁盘分区。

对于 ISO 安装

这个示例保留分区标签以**数据（data*）**开头的任何分区：

```
# coreos-installer install --ignition-url http://10.0.2.2:8080/user.ign \
      --save-partlabel 'data*' /dev/sda
```

以下示例演示了在运行 **coreos-installer** 时要保留磁盘上的第 6 个分区：

```
# coreos-installer install --ignition-url http://10.0.2.2:8080/user.ign \
      --save-partindex 6 /dev/sda
```

这个示例保留了分区 5 及更高分区：

```
# coreos-installer install --ignition-url http://10.0.2.2:8080/user.ign
      --save-partindex 5- /dev/sda
```

在前面已保存分区的示例中，**coreos-installer** 会立即重新创建分区。

对于 PXE 安装

这个 **APPEND** 选项保留分区标签以 'data'('data*')开头的所有分区：

```
coreos.inst.save_partlabel=data*
```

这个 **APPEND** 选项保留分区 5 及其后的分区：

```
coreos.inst.save_partindex=5-
```

这个 **APPEND** 选项保留分区 6:

```
coreos.inst.save_partindex=6
```

5.3.11.3.3. 标识 Ignition 配置

在进行 RHCOS 手动安装时，您可以提供两种 Ignition 配置类型，它们有不同的原因：

- **永久安装 Ignition 配置**：每个手动 RHCOS 安装都需要传递 **openshift-installer** 生成的 Ignition 配置文件之一，如 **bootstrap.ign**、**master.ign** 和 **worker.ign**，才能进行安装。

  重要

  不建议修改这些文件。

  对于 PXE 安装，您可以使用 **coreos.inst.ignition_url=** 选项在 **APPEND** 行上传递 Ignition 配置。对于 ISO 安装，在 ISO 引导至 shell 提示符后，您可以使用 **--ignition-url=** 选项在 **coreos-installer** 命令行上识别 Ignition 配置。在这两种情况下，都只支持 HTTP 和 HTTPS 协议。

- **live 安装 Ignition 配置**：此类型必须手动创建，并应该尽可能避免，因为红帽不支持它。使用此方法，Ignition 配置会传递到 live 安装介质，在引导时立即运行，并在 RHCOS 系统安装到磁盘之前和/或之后执行设置任务。这个方法只用于必须执行一次且之后不能再次应用的任务，如不能使用机器配置进行的高级分区。
  对于 PXE 或 ISO 引导，您可以创建 Ignition 配置，**APPEND ignition.config.url=** 选项，以标识 Ignition 配置的位置。您还需要附加 **ignition.firstboot ignition.platform.id=metal** 或者 **ignition.config.url** 选项。

5.3.11.3.3.1. 在 RHCOS ISO 中嵌入 Ignition 配置

您可以直接嵌入 RHCOS ISO 镜像中的 live 安装 Ignition 配置。引导 ISO 镜像后，内嵌的配置将自动应用。

流程

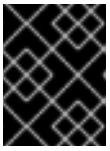
1. 从以下镜像镜像页面下载 **coreos-installer** 二进制文件：
   https://mirror.openshift.com/pub/openshift-v4/clients/coreos-installer/latest/。
2. 检索 RHCOS ISO 镜像和 Ignition 配置文件，并将其复制到可访问的目录中，如 **/mnt**:

```
# cp rhcos-<version>-live.x86_64.iso bootstrap.ign /mnt/
# chmod 644 /mnt/rhcos-<version>-live.x86_64.iso
```

3. 运行以下命令将 Ignition 配置嵌入 ISO 中：

```
# ./coreos-installer iso ignition embed -i /mnt/bootstrap.ign \
    /mnt/rhcos-<version>-live.x86_64.iso
```

   现在，您以使用该 ISO 使用指定的 live 安装 Ignition 配置来安装 RHCOS。



重要

不支持且不推荐使用 **coreos-installer iso ignition embed** 来嵌入由 **openshift-installer** 生成的文件，如 **bootstrap.ign**、**master.ign** 和 **worker.ign**。

4. 要显示嵌入的 Ignition 配置的内容并将其定向到文件中，请运行：

```
# ./coreos-installer iso ignition show /mnt/rhcos-<version>-live.x86_64.iso > mybootstrap.ign
```

```
# diff -s bootstrap.ign mybootstrap.ign
```

输出示例

```
Files bootstrap.ign and mybootstrap.ign are identical
```

5. 要删除 Ignition 配置并将 ISO 返回到其 pristine 状态（因此您可以重复使用它），请运行：

```
# ./coreos-installer iso ignition remove /mnt/rhcos-<version>-live.x86_64.iso
```

   现在，您可以将另一个 Ignition 配置嵌入到 ISO 中，或者在其 pristine 状态下使用 ISO。

5.3.11.3.4. 高级 RHCOS 安装参考

本节演示了网络配置和其他高级选项，允许您修改 Red Hat Enterprise Linux CoreOS（RHCOS）手动安装过程。下表描述了您可以与 RHCOS live installer 和 **coreos-installer** 命令一起使用的内核参数和命令行选项。

RHCOS 启动提示下的路由和绑定选项
如果从 ISO 镜像安装 RHCOS，您可以在引导该镜像时手动添加内核参数以配置节点的网络。如果没有使用网络参数，则安装默认为使用 DHCP。

重要

添加网络参数时，还必须添加 **rd.neednet=1** 内核参数。

下表描述了如何为实时 ISO 安装使用 **ip=** 、 **nameserver=** 和 **bond=** 内核参数。

注意

在添加内核参数时顺序非常重要： **ip=**，**nameserver=**，然后 **bond=**。

ISO 的路由和绑定选项

下表提供了配置 Red Hat Enterprise Linux CoreOS（RHCOS）节点网络的示例。这些是在系统引导过程中传递给 **dracut** 工具的网络选项。有关 **dracut** 支持的网络选项的详情，请参考 **dracut.cmdline** 手册页。

描述	例子
要配置一个 IP 地址，可以使用 DHCP(**ip=dhcp**)或者设置单独的静态 IP 地址(**ip=<host_ip>**)。然后在每个节点上指定 DNS 服务器 IP 地址(**nameserver=<dns_ip>**)。这个示例设置： • 节点的 IP 地址为 **10.10.10.2** • 网关地址为 **10.10.10.254** • 子网掩码为 **255.255.255.0** • 主机名为 **core0.example.com** • DNS 服务器地址为 **4.4.4.41**	ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp1s0:none nameserver=4.4.4.41
通过指定多个 **ip=** 条目来指定多个网络接口。	ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp1s0:none ip=10.10.10.3::10.10.10.254:255.255.255.0:core0.example.com:enp2s0:none

描述	例子

可选：您可以通过设置一个 **rd.route=** 值来配置到额外网络的路由。
 	
如果额外网络网关与主要网络网关不同，则默认网关必须是主要网络网关。
	配置默认网关：
	
	`ip=::10.10.10.254::::`
	
	为额外网络配置路由：
	
	`rd.route=20.20.20.0/24:20.20.20.254:enp2s0`

在单一接口中禁用 DHCP，比如当有两个或者多个网络接口时，且只有一个接口被使用。	`ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp1s0:none` `ip=::::core0.example.com:enp2s0:none`
您可以将系统中 DHCP 和静态 IP 配置与多个网络接口结合在一起。	`ip=enp1s0:dhcp` `ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp2s0:none`

可选： 您可以使用 **vlan=** 参数在单独的接口上配置 VLAN。	在网络接口中配置 VLAN 并使用静态 IP 地址：
	
	`ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp2s0.100:none`
	`vlan=enp2s0.100:enp2s0`
	
	在网络接口中配置 VLAN 并使用 DHCP：
	
	`ip=enp2s0.100:dhcp`
	`vlan=enp2s0.100:enp2s0`

您可以为每个服务器添加一个 **nameserver=** 条目来提供多个 DNS 服务器。	`nameserver=1.1.1.1` `nameserver=8.8.8.8`

可选：使用 **bond=** 选项支持将多个网络接口绑定到一个接口。在这两个示例中：
 	
• 配置绑定接口的语法为：**bond=name[:network_interfaces][:options]**
• *name* 是绑定设备名称（**bond0**），*network_interfaces* 代表用逗号分开的物理（以太网）接口(**em1,em2**)的列表，*options* 是用逗号分开的绑定选项列表。输入 **modinfo bonding** 查看可用选项。
• 当使用 **bond=** 创建绑定接口时，您必须指定如何分配 IP 地址以及绑定接口的其他信息。
	要将绑定的接口配置为使用 DHCP，请将绑定的 IP 地址设置为 **dhcp**。例如：
	
	`bond=bond0:em1,em2:mode=active-backup`
	`ip=bond0:dhcp`
	
	要将绑定接口配置为使用静态 IP 地址，请输入您需要的特定 IP 地址以及相关信息。例如：
	
	`bond=bond0:em1,em2:mode=active-backup`
	`ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:bond0:none`

描述	例子
可选： 您可以使用 **vlan=** 参数在绑定接口上配置 VLAN。	使用 VLAN 配置绑定接口并使用 DHCP： ip=bond0.100:dhcp bond=bond0:em1,em2:mode=active-backup vlan=bond0.100:bond0 使用 VLAN 配置绑定接口，并使用静态 IP 地址： ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:bond0.100:none bond=bond0:em1,em2:mode=active-backup vlan=bond0.100:bond0
可选：通过使用 **team=** 参数，网络合作可用作绑定的替代选择。在此例中： • 配置组接口的语法为：**team=name[:network_interfaces]** *name* 是组设备名称(**team0**)，**network_interfaces** 代表以逗号分隔的物理（以太网）接口（**em1**、**em2**）列表。  **注意** 当 RHCOS 切换到即将推出的 RHEL 版本时，团队计划会被弃用。如需更多信息，请参阅红帽知识库文章。	配置网络团队： team=team0:em1,em2 ip=team0:dhcp

coreos.inst 引导选项用于 ISO 或 PXE 安装

虽然您可以将大多数标准安装引导参数传递给 live 安装程序，但也有一些特定于 RHCOS live 安装程序的参数。

- 对于 ISO，可以通过中断 RHCOS 安装程序来添加这些选项。
- 对于 PXE 或 iPXE，这些选项必须在启动 PXE 内核前添加到 **APPEND** 行中。您无法中断实时 PXE 安装。

下表显示了用于 ISO 和 PXE 安装的 RHCOS live installer 引导选项。

表 5.40. coreos.inst 引导选项

参数	描述
coreos.inst.install_dev	必需。要安装的系统中的块设备。虽然可以使用 **sda** 这样的相对路径，但建议使用完整路径，如 **/dev/sda**。

参数	描述
coreos.inst.ignition_url	可选：嵌入到已安装系统中的 Ignition 配置的 UR如果没有指定 URL，则不会嵌入 Ignition 配置。
coreos.inst.save_partlabel	可选：在安装过程中要保留的分区压缩标签。允许使用 glob 风格的通配符。指定分区不需要存在。
coreos.inst.save_partindex	可选：在安装过程中完成要保留的分区的分离索引。可以使用 **m-n** 指定范围，**m** 或 **n** 可以被省略。指定分区不需要存在。
coreos.inst.insecure	可选：将 **coreos.inst.image_url** 指定的 OS 镜像提交取消签名。
coreos.inst.image_url	可选：下载并安装指定的 RHCOS 镜像。 ● 这个参数不应该在生产环境中使用，而是只用于调试目的。 ● 虽然在 RHCOS 的安装版本与 live 介质的版本不匹配时可以使用这个参数，但建议使用与您要安装版本匹配的介质。 ● 如果您使用的是 **coreos.inst.image_url**。还必须使用 **coreos.inst.insecure**。这是因为，裸机介质没有为 OpenShift Container Platform 进行 GPG 签名。 ● 只支持 HTTP 和 HTTPS 协议。
coreos.inst.skip_reboot	可选：安装后该系统不会重启。安装完成后，您会收到提示，提示您检查在安装过程中发生的情况。这个参数不应该在生产环境中使用，而是只用于调试目的。
coreos.inst.platform_id	可选：安装 RHCOS 镜像的平台的 Ignition 平台 ID。默认为 **metal**。这个选项决定是否从云供应商（如 VMware）请求 Ignition 配置。例如：**coreos.inst.platform_id=vmware**。
ignition.config.url	可选：用于实时启动的 Ignition 配置的 URL。例如，它可以用来定制调用 **coreos-installer** 的方式，或者用来在安装前或安装后运行代码。这与 **coreos.inst.ignition_url**（这是已安装系统的 Ignition 配置）不同。

ISO 安装的 coreos-installer 选项

您还可以直接从命令行调用 **coreos-installer** 命令来安装 RHCOS。上表中的内核参数提供了在引导时自动调用 **coreos-installer** 的快捷方式，但您可以在 shell 提示符运行时将类似的参数直接传递给 **coreos-installer**。

下表显示了您可以在实时安装过程中从 shell 提示符传递给 **coreos-installer** 命令的选项和子命令。

表 5.41. CoreOS-installer 命令行选项、参数和子命令

命令行选项	
选项	**描述**
-u, **--image-url <url>**	手动指定镜像 URL。
-f, **--image-file <path>**	手动指定本地镜像文件。
-i, --ignition-file <path>	从文件中嵌入 Ignition 配置。
-I, **--ignition-url <URL>**	从 URL 嵌入 Ignition 配置。
--ignition-hash <digest>	Ignition config 的 **type-value** 的文摘值。
-p, **--platform <name>**	覆盖 Ignition 平台 ID。
--append-karg <arg>…	附加默认内核参数。
--delete-karg <arg>…	删除默认内核参数。
-n, **--copy-network**	从安装环境中复制网络配置。  **重要** **copy-network** 选项只复制 **/etc/NetworkManager/system-connections** 下的网络配置。特别是，它不会复制系统主机名。
--network-dir <path>	使用 **-n**。默认为 **/etc/NetworkManager/system-connections/**。
--save-partlabel <lx>..	使用这个标签 glob 保存分区。
--save-partindex <id>…	使用这个数值或者范围保存分区。
--offline	强制离线安装。
--insecure	跳过签名验证。
--insecure-ignition	允许没有 HTTPS 或 hash 的 Ignition URL。
--architecture <name>	目标 CPU 架构。默认为 **x86_64**。
--preserve-on-error	出现错误时不清除分区表。

-h，--help	打印帮助信息。
命令行参数	
参数	**描述**
<device>	目的设备。
CoreOS-installer 嵌入的 Ignition 命令	
命令	**描述**
$ coreos-installer iso ignition embed <options> --ignition-file <file_path> <ISO_image>	在 ISO 镜像中嵌入 Ignition 配置。
coreos-installer iso ignition show <options> <ISO_image>	显示来自 ISO 镜像的内嵌 Ignition 配置。
coreos-installer iso ignition remove <options> <ISO_image>	从 ISO 镜像中删除嵌入的 Ignition 配置。
coreos-installer ISO Ignition 选项	
选项	**描述**
-f, --force	覆盖现有的 Ignition 配置。
-i, --ignition-file <path>	要使用的 Ignition 配置。默认为 **stdin**。
-o, --output <path>	将 ISO 写入到一个新输出文件。
-h，--help	打印帮助信息。
coreos-installer PXE Ignition 命令	
命令	**描述**
请注意，不是所有子命令都接受这些选项。	
coreos-installer pxe ignition wrap <options>	在镜像中嵌套 Ignition 配置。
coreos-installer pxe ignition unwrap <options> <image_name>	显示在镜像中嵌套的 Ignition 配置。
coreos-installer pxe ignition unwrap <options> <initrd_name>	在 **initrd** 镜像中显示嵌套的 Ignition 配置。

coreos-installer PXE Ignition 选项	
选项	**描述**
-i, --ignition-file <path>	要使用的 Ignition 配置。默认为 **stdin**。
-o, --output <path>	将 ISO 写入到一个新输出文件。
-h, --help	打印帮助信息。

5.3.12. 创建集群

要创建 OpenShift Container Platform 集群，请等待您通过安装程序生成的 Ignition 配置文件所置备的机器上完成 bootstrap 过程。

先决条件

- 为集群创建所需的基础架构。
- 已获得安装程序并为集群生成了 Ignition 配置文件。
- 已使用 Ignition 配置文件为集群创建 RHCOS 机器。

流程

1. 监控 bootstrap 过程：

```
$ ./openshift-install --dir <installation_directory> wait-for bootstrap-complete \ 1
    --log-level=info 2
```

1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

2 要查看不同的安装详情，请指定 **warn**、**debug** 或 **error**，而不要指定 **info**。

输出示例

```
INFO Waiting up to 30m0s for the Kubernetes API at https://api.test.example.com:6443...
INFO API v1.19.0 up
INFO Waiting up to 30m0s for bootstrapping to complete...
INFO It is now safe to remove the bootstrap resources
```

Kubernetes API 服务器提示已在 control plane 机器上完成 bootstrap 时，命令运行成功。

2. bootstrap 过程完成后，请从负载均衡器中删除 bootstrap 机器。

重要

此时您必须从负载均衡器中删除 bootstrap 机器。您还可以删除或重新格式化机器本身。

5.3.13. 使用 CLI 登录到集群

您可以通过导出集群 **kubeconfig** 文件，以默认系统用户身份登录集群。**kubeconfig** 文件包含关于集群的信息，供 CLI 用于将客户端连接到正确集群和 API 服务器。该文件特只适用于一个特定的集群，在 OpenShift Container Platform 安装过程中创建。

先决条件

- 已部署了 OpenShift Container Platform 集群。
- 已安装 **oc** CLI。

流程

1. 导出 **kubeadmin** 凭证：

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

2. 使用导出的配置，验证能否成功运行 **oc** 命令：

```
$ oc whoami
```

输出示例

```
system:admin
```

5.3.14. 批准机器的证书签名请求

将机器添加到集群时，会为您添加的每台机器生成两个待处理证书签名请求（CSR）。您必须确认这些 CSR 已获得批准，或根据需要自行批准。客户端请求必须首先被批准，然后是服务器请求。

先决条件

- 您已将机器添加到集群中。

流程

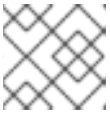
1. 确认集群可以识别这些机器：

```
$ oc get nodes
```

输出示例

```
NAME     STATUS   ROLES   AGE  VERSION
master-0 Ready    master  63m  v1.19.0
master-1 Ready    master  63m  v1.19.0
master-2 Ready    master  64m  v1.19.0
```

输出将列出您创建的所有机器。



注意

在一些 CSR 被批准前，以上输出可能不包括计算节点（也称为 worker 节点）。

2. 检查待处理的 CSR，并确保可以看到添加到集群中的每台机器都有 **Pending** 或 **Approved** 状态的客户端请求：

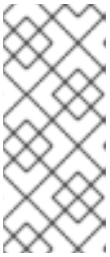
```
$ oc get csr
```

输出示例

```
NAME        AGE    REQUESTOR                                                   CONDITION
csr-8b2br   15m    system:serviceaccount:openshift-machine-config-operator:node-bootstrapper   Pending
csr-8vnps   15m    system:serviceaccount:openshift-machine-config-operator:node-bootstrapper   Pending
...
```

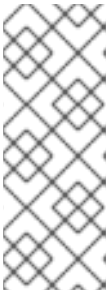
在本例中，两台机器加入了集群。您可能会在列表中看到更多已批准的 CSR。

3. 如果 CSR 没有获得批准，请在所添加机器的所有待处理 CSR 都处于 **Pending** 状态后，为您的集群机器批准这些 CSR：



注意

由于 CSR 会自动轮转，因此请在将机器添加到集群后一小时内批准您的 CSR。如果没有在一小时内批准，证书将会轮转，每个节点将会存在多个证书。您必须批准所有这些证书。批准客户端 CSR 后， Kubelet 为服务证书创建辅助 CSR，这需要手动批准。然后，如果 Kubelet 请求具有相同参数的新证书，则 **machine-approver** 会自动批准后续服务证书续订请求。



注意

对于在未启用机器 API 的平台中运行的集群，如裸机和其他用户置备的基础架构，必须采用一种方法自动批准 kubelet 提供证书请求（CSR）。如果没有批准请求，则 **oc exec**、**oc rsh** 和 **oc logs** 命令将无法成功，因为 API 服务器连接到 kubelet 时需要服务证书。与 Kubelet 端点联系的任何操作都需要此证书批准。这个方法必须监视新的 CSR，确认 CSR 由 **system:node** 或 **system:admin** 组中的 **node-bootstrapper** 服务帐户提交，并确认节点的身份。

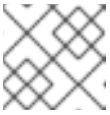
- 若要单独批准，请对每个有效的 CSR 运行以下命令：

```
$ oc adm certificate approve <csr_name> 1
```

1 **<csr_name>** 是当前 CSR 列表中 CSR 的名称。

- 要批准所有待处理的 CSR，请运行以下命令：

```
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}{{"\n"}}{{end}}{{end}}' | xargs --no-run-if-empty oc adm certificate approve
```



注意

在有些 CSR 被批准前，一些 Operator 可能无法使用。

4. 现在，您的客户端请求已被批准，您必须查看添加到集群中的每台机器的服务器请求：

```
$ oc get csr
```

输出示例

```
NAME        AGE     REQUESTOR                                                   CONDITION
csr-bfd72   5m26s   system:node:ip-10-0-50-126.us-east-2.compute.internal       Pending
csr-c57lv   5m26s   system:node:ip-10-0-95-157.us-east-2.compute.internal       Pending
...
```

5. 如果剩余的 CSR 没有被批准，且处于 **Pending** 状态，请批准集群机器的 CSR：

- 若要单独批准，请对每个有效的 CSR 运行以下命令：

```
$ oc adm certificate approve <csr_name> 1
```

1 **<csr_name>** 是当前 CSR 列表中 CSR 的名称。

- 要批准所有待处理的 CSR，请运行以下命令：

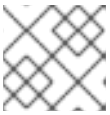
```
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}{{"\n"}}
{{end}}{{end}}' | xargs oc adm certificate approve
```

6. 批准所有客户端和服务器 CSR 后，器将处于 **Ready** 状态。运行以下命令验证：

```
$ oc get nodes
```

输出示例

```
NAME      STATUS    ROLES   AGE  VERSION
master-0  Ready     master  73m  v1.20.0
master-1  Ready     master  73m  v1.20.0
master-2  Ready     master  74m  v1.20.0
worker-0  Ready     worker  11m  v1.20.0
worker-1  Ready     worker  11m  v1.20.0
```



注意

批准服务器 CSR 后可能需要几分钟时间让机器转换为 **Ready** 状态。

其他信息

- 如需有关 CSR 的更多信息，请参阅证书签名请求。

5.3.15. 初始 Operator 配置

在 control plane 初始化后，您必须立即配置一些 Operator 以便它们都可用。

先决条件

- 您的 control plane 已初始化。

流程

1. 观察集群组件上线：

```
$ watch -n5 oc get clusteroperators
```

输出示例

```
NAME                                       VERSION AVAILABLE   PROGRESSING   DEGRADED
SINCE
authentication                             4.6.0   True        False         False      3h56m
cloud-credential                           4.6.0   True        False         False      29h
cluster-autoscaler                         4.6.0   True        False         False      29h
config-operator                            4.6.0   True        False         False      6h39m
console                                    4.6.0   True        False         False      3h59m
csi-snapshot-controller                    4.6.0   True        False         False      4h12m
dns                                        4.6.0   True        False         False      4h15m
etcd                                       4.6.0   True        False         False      29h
image-registry                             4.6.0   True        False         False      3h59m
ingress                                    4.6.0   True        False         False      4h30m
insights                                   4.6.0   True        False         False      29h
kube-apiserver                             4.6.0   True        False         False      29h
kube-controller-manager                    4.6.0   True        False         False      29h
kube-scheduler                             4.6.0   True        False         False      29h
kube-storage-version-migrator              4.6.0   True        False         False      4h2m
machine-api                                4.6.0   True        False         False      29h
machine-approver                           4.6.0   True        False         False      6h34m
machine-config                             4.6.0   True        False         False      3h56m
marketplace                                4.6.0   True        False         False      4h2m
monitoring                                 4.6.0   True        False         False      6h31m
network                                    4.6.0   True        False         False      29h
node-tuning                                4.6.0   True        False         False      4h30m
openshift-apiserver                        4.6.0   True        False         False      3h56m
openshift-controller-manager               4.6.0   True        False         False      4h36m
openshift-samples                          4.6.0   True        False         False      4h30m
operator-lifecycle-manager                 4.6.0   True        False         False      29h
operator-lifecycle-manager-catalog         4.6.0   True        False         False      29h
operator-lifecycle-manager-packageserver   4.6.0   True        False         False      3h59m
service-ca                                 4.6.0   True        False         False      29h
storage                                    4.6.0   True        False         False      4h30m
```

2. 配置不可用的 Operator。

5.3.15.1. 禁用默认的 OperatorHub 源

在 OpenShift Container Platform 安装过程中，默认为 OperatorHub 配置由红帽和社区项目提供的源内容的 operator 目录。在受限网络环境中，必须以集群管理员身份禁用默认目录。

流程

- 通过在 **OperatorHub** 对象中添加 **disableAllDefaultSources: true** 来禁用默认目录的源：

```
$ oc patch OperatorHub cluster --type json \
    -p '[{"op": "add", "path": "/spec/disableAllDefaultSources", "value": true}]'
```

提示

或者，您可以使用 Web 控制台管理目录源。在 **Administration → Cluster Settings → Global Configuration → OperatorHub** 页面中，点 **Sources** 选项卡，其中可创建、删除、禁用和启用单独的源。

5.3.15.2. 镜像 registry 存储配置

对于不提供默认存储的平台，Image Registry Operator 最初将不可用。安装后，您必须配置 registry 使用的存储，这样 Registry Operator 才可用。

示配置生产集群所需的持久性卷的说明。如果适用，显示有关将空目录配置为存储位置的说明，该位置只可用于非生产集群。

另外还提供了在升级过程中使用 **Recreate** rollout 策略来允许镜像 registry 使用块存储类型的说明。

5.3.15.2.1. 更改镜像 registry 的管理状态

要启动镜像 registry，需要把 Image Registry Operator 配置的 **managementState** 从 **Removed** 改为 **Managed**。

流程

- 将 **managementState** Image Registry Operator 配置从 **Removed** 改为 **Managed**。例如：

```
$ oc patch configs.imageregistry.operator.openshift.io cluster --type merge --patch '{"spec":
{"managementState":"Managed"}}'
```

5.3.15.2.2. 为裸机和其他手动安装配置 registry 存储

作为集群管理员，在安装后需要配置 registry 来使用存储。

先决条件

- 具有 Cluster Administrator 权限
- 使用手动置备的 Red Hat Enterprise Linux CoreOS（RHCOS）节点（如裸机）的集群。
- 为集群置备的持久性存储，如 Red Hat OpenShift Container Storage。

重要

如果您只有一个副本，OpenShift Container Platform 支持对镜像 registry 存储的 **ReadWriteOnce** 访问。要部署支持高可用性的、带有两个或多个副本的镜像 registry，需要 **ReadWriteMany** 访问设置。

- 必须具有 100Gi 容量。

流程

1. 为了配置 registry 使用存储，需要修改 **configs.imageregistry/cluster** 资源中的 **spec.storage.pvc** 。

注意

使用共享存储时，请查看您的安全设置以防止被外部访问。

2. 验证您没有 registry pod:

```
$ oc get pod -n openshift-image-registry
```

注意

如果存储类型为 **emptyDIR**，则副本数不能超过 **1**。

3. 检查 registry 配置：

```
$ oc edit configs.imageregistry.operator.openshift.io
```

输出示例

```
storage:
  pvc:
    claim:
```

将 **claim** 字段留空以允许自动创建一个 **image-registry-storage** PVC。

4. 检查 **clusteroperator** 的状态：

```
$ oc get clusteroperator image-registry
```

5. 确保您的 registry 设置为 manage，以启用镜像的构建和推送。

- 运行：

```
$ oc edit configs.imageregistry/cluster
```

然后将行改

```
managementState: Removed
```

为

```
managementState: Managed
```

5.3.15.2.3. 在非生产集群中配置镜像 registry 存储

您必须为 Image Registry Operator 配置存储。对于非生产集群，您可以将镜像 registry 设置为空目录。如果您这样做，重启 registry 后会丢失所有镜像。

流程

- 将镜像 registry 存储设置为空目录：

```
$ oc patch configs.imageregistry.operator.openshift.io cluster --type merge --patch '{"spec":{"storage":{"emptyDir":{}}}}'
```

> **警告**
>
> 仅可为非生产集群配置这个选项。

如果在 Image Registry Operator 初始化其组件前运行此命令，**oc patch** 命令会失败并显示以下错误：

```
Error from server (NotFound): configs.imageregistry.operator.openshift.io "cluster" not found
```

等待几分钟，然后再次运行该命令。

5.3.15.2.4. 配置块 registry 存储

要允许镜像 registry 在作为集群管理员升级过程中使用块存储类型，您可以使用 **Recreate** rollout 策略。

> **重要**
>
> 支持块存储卷，但不建议将其与生产环境中的镜像 registry 一起使用。在块存储上配置 registry 的安装不具有高可用性，因为 registry 无法拥有多个副本。

流程

1. 要将镜像 registry 存储设置为块存储类型，对 registry 进行补丁，使其使用 **Recreate** rollout 策略，并只使用一个（**1**）副本运行：

```
$ oc patch config.imageregistry.operator.openshift.io/cluster --type=merge -p '{"spec":{"rolloutStrategy":"Recreate","replicas":1}}'
```

2. 为块存储设备置备 PV，并为该卷创建 PVC。请求的块卷使用 ReadWriteOnce（RWO）访问模式。

3. 编辑 registry 配置，使其引用正确的 PVC。

5.3.16. 在用户置备的基础架构上完成安装

完成 Operator 配置后，可以在您提供的基础架构上完成集群安装。

先决条件

- 您的 control plane 已初始化。
- 已完成初始 Operator 配置。

流程

1. 使用以下命令确认所有集群组件都已在线：

```
$ watch -n5 oc get clusteroperators
```

输出示例

```
NAME                                       VERSION AVAILABLE   PROGRESSING   DEGRADED
SINCE
authentication                             4.6.0   True        False         False      3h56m
cloud-credential                           4.6.0   True        False         False      29h
cluster-autoscaler                         4.6.0   True        False         False      29h
config-operator                            4.6.0   True        False         False      6h39m
console                                    4.6.0   True        False         False      3h59m
csi-snapshot-controller                    4.6.0   True        False         False      4h12m
dns                                        4.6.0   True        False         False      4h15m
etcd                                       4.6.0   True        False         False      29h
image-registry                             4.6.0   True        False         False      3h59m
ingress                                    4.6.0   True        False         False      4h30m
insights                                   4.6.0   True        False         False      29h
kube-apiserver                             4.6.0   True        False         False      29h
kube-controller-manager                    4.6.0   True        False         False      29h
kube-scheduler                             4.6.0   True        False         False      29h
kube-storage-version-migrator              4.6.0   True        False         False      4h2m
machine-api                                4.6.0   True        False         False      29h
machine-approver                           4.6.0   True        False         False      6h34m
machine-config                             4.6.0   True        False         False      3h56m
marketplace                                4.6.0   True        False         False      4h2m
monitoring                                 4.6.0   True        False         False      6h31m
network                                    4.6.0   True        False         False      29h
node-tuning                                4.6.0   True        False         False      4h30m
openshift-apiserver                        4.6.0   True        False         False      3h56m
openshift-controller-manager               4.6.0   True        False         False      4h36m
openshift-samples                          4.6.0   True        False         False      4h30m
operator-lifecycle-manager                 4.6.0   True        False         False      29h
operator-lifecycle-manager-catalog         4.6.0   True        False         False      29h
operator-lifecycle-manager-packageserver   4.6.0   True        False         False      3h59m
service-ca                                 4.6.0   True        False         False      29h
storage                                    4.6.0   True        False         False      4h30m
```

或者，通过以下命令，如果所有集群都可用您会接到通知。它还检索并显示凭证：

```
$ ./openshift-install --dir <installation_directory> wait-for install-complete 1
```

1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

输出示例

```
INFO Waiting up to 30m0s for the cluster to initialize...
```

Cluster Version Operator 完成从 Kubernetes API 服务器部署 OpenShift Container Platform 集群时，命令运行成功。

重要

- 安装程序生成的 Ignition 配置文件包含在 24 小时后过期的证书，然后在过期时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外情况是，您需要手动批准待处理的 **node-bootstrapper** 证书签名请求（CSR）来恢复 kubelet 证书。如需更多信息，请参阅*从过期的 control plane 证书中恢复*的文档。
- 建议您在生成 12 小时后使用 Ignition 配置文件，因为集群安装后 24 小时证书从 16 小时轮转至 22 小时。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中运行证书更新时避免安装失败。

2. 确认 Kubernetes API 服务器正在与 pod 通信。

   a. 要查看所有 pod 的列表，请使用以下命令：

   ```
   $ oc get pods --all-namespaces
   ```

   输出示例

   ```
   NAMESPACE                       NAME                                         READY   STATUS
   RESTARTS   AGE
   openshift-apiserver-operator    openshift-apiserver-operator-85cb746d55-zqhs8   1/1
   Running    1          9m
   openshift-apiserver             apiserver-67b9g                               1/1     Running     0
   3m
   openshift-apiserver             apiserver-ljcmx                               1/1     Running     0
   1m
   openshift-apiserver             apiserver-z25h4                               1/1     Running     0
   2m
   openshift-authentication-operator authentication-operator-69d5d8bf84-vh2n8       1/1
   Running    0          5m
   ...
   ```

   b. 使用以下命令，查看上一命令的输出中所列 pod 的日志：

   ```
   $ oc logs <pod_name> -n <namespace>
   ```
   1

   1 指定 pod 名称和命名空间，如上一命令的输出中所示。

   如果 pod 日志显示，Kubernetes API 服务器可以与集群机器通信。

3. 在 Cluster registration 页面注册您的集群。

5.3.17. OpenShift Container Platform 的 Telemetry 访问

在 OpenShift Container Platform 4.6 中，默认运行的 Telemetry 服务提供有关集群健康状况和成功更新的指标，需要访问互联网。如果您的集群连接到互联网，Telemetry 会自动运行，而且集群会注册到 OpenShift Cluster Manager。

确认 OpenShift Cluster Manager 清单正确后，可以由 Telemetry 自动维护，也可以使用 OpenShift Cluster Manager 手动维护，使用订阅监控来跟踪帐户或多集群级别的 OpenShift Container Platform 订阅。

其他资源

- 有关 Telemetry 服务的更多信息，请参阅关于远程健康监控。

5.3.18. 后续步骤

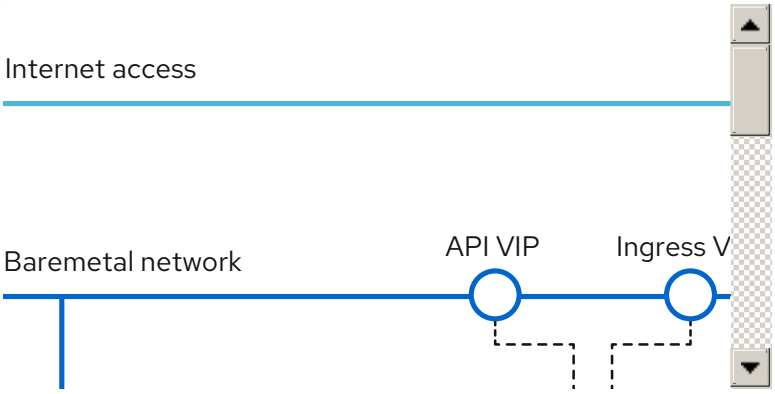
- 自定义集群。
- 为 Cluster Samples Operator 和 **must-gather** 工具配置镜像流。
- 了解如何在受限网络中使用 Operator Lifecycle Manager（OLM）。
- 如果您用来安装集群的镜像 registry 具有一个可信任的 CA，通过配置额外的信任存储将其添加到集群中。
- 如果需要，您可以选择不使用远程健康报告。

第 6 章 在裸机上部署安装程序置备的集群

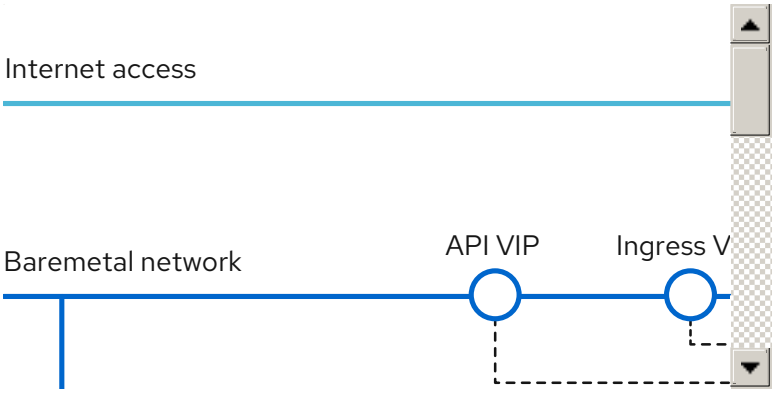
6.1. 概述

安装程序置备的安装支持在裸机节点上安装 OpenShift Container Platform。本指南提供了一种成功安装的方法。

在裸机上的安装程序置备安装过程中，裸机节点上标记为 **provisioner** 的安装程序会创建一个 bootstrap 虚拟机。bootstrap 虚拟机的角色是协助部署 OpenShift Container Platform 集群。bootstrap 虚拟机通过网桥接连接到 **baremetal** 网络和 **provisioning** 网络（如果存在）。



当完成 OpenShift Container Platform control plane 节点安装并可以正常工作时，安装程序会自动销毁 bootstrap 虚拟机，并相应地将虚拟 IP 地址（VIP）移到适当的节点。API VIP 移至 control plane 节点，Ingress VIP 移至 worker 节点。



6.2. 先决条件

OpenShift Container Platform 安装程序置备的安装需要：

1. 安装了 Red Hat Enterprise Linux（RHEL）8.x 的一个置备程序节点。
2. 三个 control plane 节点。
3. 对每个节点的 Baseboard Management Controller（BMC）访问。
4. 至少一个网络：
    a. 一个**必需的**可路由网络
    b. 一个用于置备节点的**可选**网络 ; 和

c. 一个**可选的**管理网络。

在开始 OpenShift Container Platform 安装程序置备的安装前，请确保硬件环境满足以下要求。

6.2.1. 节点要求

安装程序置备的安装有多个硬件节点要求：

- **CPU 架构：** 所有节点都必须使用 **x86_64** CPU 架构。
- **类似的节点：** 红帽建议每个角色都有相同的配置。也就是说，红帽推荐节点具有相同的品牌和型号，它们具有相同的 CPU、内存和存储配置。
- **Baseboard Management Controller: provisioner** 节点必须能够访问每个 OpenShift Container Platform 集群节点的基板管理控制器（BMC）。您可以使用 IPMI、RedFish 或一个私有协议。
- **最新一代：** 节点必须是最新一代。因为安装程序置备的安装依赖于 BMC 协议，所以硬件必须支持 IPMI 密码套件 17。另外，RHEL 8 附带了 RAID 控制器的最新驱动程序。确保节点足以支持 **provisioner** 节点运行 RHEL 8，支持 control plane 和 worker 节点运行 RHCOS 8。
- **registry 节点：**（可选）如果设置了一个断开连接的 mirrored registry，建议 registry 驻留在自己的节点上。
- **provisioner 节点：** 安装程序置备的安装需要一个 **provisioner** 节点。
- **Control plane:** 安装程序置备的安装需要三个 control plane 节点才能进行高可用性。
- **Worker 节点：** 虽然不需要，但典型的生产环境集群具有一个或多个 worker 节点。较小的集群在开发和测试过程中为管理员和开发人员提供更多资源。
- **网络接口：** 每个节点必须至少有一个网络接口用于可路由的 **baremetal** 网络。在使用 **provisioning** 网络进行部署时，每个节点都必须有一个网络接口用于 **provisioning** 网络。使用 **provisioning** 网络是默认配置。对于 provisioning 网络，网络接口命名必须一致的 control plane 节点。例如，如果 control plane 节点将 **eth0** NIC 用于 provisioning 网络，则其他 control plane 节点也必须使用它。
- **统一的可扩展固件接口（UEFI）**：在 **provisioning** 网络中使用 IPv6 时，,安装程序置备的安装需要在所有 OpenShift Container Platform 节点上进行 UEFI 引导。另外，UEFI Device PXE Settings 必需被设置为在 **provisioning** 网络 NIC 中使用 IPv6 协议，但 **忽略 provisioning 网络会删除这个要求。**

6.2.2. 网络要求

OpenShift Container Platform 安装程序置备的安装默认涉及多个网络要求。首先，安装程序置备的安装涉及一个不可路由的 **provisioning** 网络，用于在每个裸机节点上置备操作系统，以及一个可路由的 **baremetal** 网络。因为安装程序置备的安装会部署 **ironic-dnsmasq**，所以网络不能有在同一广播域中运行的其他 DHCP 服务器。网络管理员必须为 OpenShift Container Platform 集群中的每个节点保留 IP 地址。

网络时间协议（NTP）

建议集群中的每个 OpenShift Container Platform 节点都可以访问可使用 DHCP 发现的网络时间协议（NTP）服务器。虽然在没有 NTP 服务器的情况下安装是可能的，异步服务器时钟可能会导致错误。使用 NTP 服务器可以防止这个问题。

配置 NIC

OpenShift Container Platform 使用两个网络部署：

- **provisioning**：**provisioning** 网络是一个 **可选的**、不可路由的网络，用于在作为 OpenShift Container Platform 集群一部分的每个节点上置备底层操作系统。当使用 **provisioning** 网络进行部署时，每个节点上的第一个 NIC（如 **eth0** 或 **eno1** ） **必须**是与 **provisioning** 网络的接口。
- **baremetal**：**baremetal** 网络是一个可路由的网络。当使用 **provisioning** 网络部署时，每个节点的第二个 NIC（如 **eth1** 或 **eno2**）**必须**是与 **baremetal** 网络的接口。当在没有 **provisioning** 网络的情况下部署时，您可以使用每个节点上的任意 NIC 作为与 **provisioning** 的接口。

重要

每个 NIC 应该位于与适当网络对应的独立 VLAN 中。

配置 DNS 服务器

客户端通过 **baremetal** 网络访问 OpenShift Container Platform 集群节点。网络管理员必须配置子域或子区，其中 CN 扩展是集群名称。

```
<cluster-name>.<domain-name>
```

例如：

```
test-cluster.example.com
```

使用 DHCP 服务器为节点保留 IP 地址

对于 **baremetal** 网络，网络管理员必须保留一组 IP 地址，其中包括：

1. 两个虚拟 IP 地址。
   - API 端点的一个 IP 地址
   - 一个用于通配符入口端点的 IP 地址
2. 一个用于 provisioner 节点的 IP 地址。
3. 每个 control plane（master）节点有一个 IP 地址。
4. 每个 worker 节点一个 IP 地址（如果适用）。

下表提供了一个完全限定域名的示范性实施API 和 Nameserver 地址以规范名称（canonical name）扩展开头。control plane 和 worker 节点的主机名只是示例，您可以使用您喜欢的任何主机命名规则。

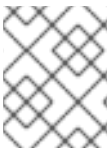
使用	主机名	IP
API	*api.<cluster-name>.<domain>*	*<ip>*
Ingress LB (apps)	**.apps.<cluster-name>.<domain>*	*<ip>*
Provisioner node	*provisioner.<cluster-name>.<domain>*	*<ip>*
Master-0	*openshift-master-0.<cluster-name>.<domain>*	*<ip>*

使用	主机名	IP
Master-1	*openshift-master-1.<cluster-name>-.<domain>*	*<ip>*
Master-2	*openshift-master-2.<cluster-name>.<domain>*	*<ip>*
Worker-0	*openshift-worker-0.<cluster-name>.<domain>*	*<ip>*
Worker-1	*openshift-worker-1.<cluster-name>.<domain>*	*<ip>*
Worker-n	*openshift-worker-n.<cluster-name>.<domain>*	*<ip>*

无 provisioning 网络的额外要求

所有安装程序置备的安装都需要一个 **baremetal** 网络。**baremetal** 网络是一个可路由的网络，用于外部网络访问外部世界。除了向 OpenShift Container Platform 集群节点提供的 IP 地址外，没有 **provisioning** 网络的安装还需要以下：

- 在 **install-config.yaml** 配置文件中将来自 **baremetal** 网络中一个可用 IP 地址设置为 **bootstrapProvisioningIP** 配置设置。
- 在 **install-config.yaml** 配置文件中将来自 **baremetal** 网络中的一个可用 IP 地址设置为 **provisioningHostIP** 配置设置。
- 使用 RedFish Virtual Media/iDRAC Virtual Media 部署 OpenShift Container Platform 集群。



注意

当使用 **provisioning** 网络时，不需要为 **bootstrapProvisioningIP** 和 **provisioningHostIP** 配置额外的 IP 地址。

带外管理 IP 地址的端口访问

带外管理 IP 地址位于与节点独立的网络上。为确保带外管理可以在安装期间与 **裸机** 节点通信，必须授予带外管理 IP 地址对 TCP 6180 端口的访问权限。

6.2.3. 配置节点

在使用 provisioning 网络时配置节点

集群中的每个节点都需要以下配置才能正确安装。

警告

如果在节点间不匹配则会导致安装失败。

虽然集群节点可以包含多于 2 个 NIC，但安装过程只关注于前两个 NIC:

NIC	网络	VLAN
NIC1	**provisioning**	<provisioning-vlan>
NIC2	**baremetal**	<baremetal-vlan>

NIC1 是一个不可路由的网络（**provisioning**），仅用于安装 OpenShift Container Platform 集群。

置备程序节点上的 Red Hat Enterprise Linux（RHEL）8.x 安装过程可能会有所不同。要使用本地 Satellite 服务器或者 PXE 服务器安装 Red Hat Enterprise Linux（RHEL）8.x，PXE 启用 NIC2。

PXE	引导顺序
NIC1 PXE-enabled **provisioning** 网络	1
NIC2 **baremetal** 网络。启用 PXE 是可选的。	2

注意

确保在所有其他 NIC 中禁用 PXE。

配置 control plane 和 worker 节点，如下所示：

PXE	引导顺序
NIC1 PXE-enabled（provisioning 网络）	1

在没有 provisioning 网络的情况下配置节点

安装过程需要一个 NIC：

NIC	网络	VLAN
NICx	**baremetal**	<baremetal-vlan>

NICx 是一个可路由的网络（**baremetal**），它用于安装 OpenShift Container Platform 集群，并可路由到互联网。

6.2.4. 带外管理

节点通常还会有一个额外的、由 Baseboard Management Controller（BMC）使用的 NIC。这些 BMC 必须可以通过 **provisioner** 节点访问。

每个节点需要可以通过带外管理进行访问。在使用带外管理网络时，**provisioner** 节点需要访问带外管理网络才能成功安装 OpenShift Container Platform 4。

带外管理设置已超出本文档的范围。我们建议单独设置一个带外管理网络。但是，使用 **provisioning** 网络或 **baremetal** 网络是有效的选项。

6.2.5. 安装所需的数据

在安装 OpenShift Container Platform 集群前，从所有集群节点收集以下信息：

- 带外管理 IP
  - 示例
    - Dell (iDRAC) IP
    - HP (iLO) IP

使用 provisioning 网络时

- NIC1 (**provisioning**) MAC 地址
- NIC2 (**baremetal**) MAC 地址

在省略 provisioning 网络时

- NICx (**baremetal**) MAC 地址

6.2.6. 节点验证清单

使用 provisioning 网络时

❑ 为 **provisioning** 网络配置了 NIC1 VLAN。

❑ 为 **baremetal** 网络配置了 NIC2 VLAN。

❑ 在 provisioner、control plane (master) 和 worker 节点上，NIC1 启用了 PXE。

❑ PXE 在所有其他 NIC 上都被禁用。

❑ 配置了 control plane 和 worker 节点。

❑ 所有节点都可以通过带外管理访问。

❑ 已创建一个单独的管理网络（可选）

❑ 安装所需的数据。

在省略 provisioning 网络时

❑ 为 **baremetal** 网络配置 NICx VLAN。

❑ 配置了 control plane 和 worker 节点。

❑ 所有节点都可以通过带外管理访问。

❑ 已创建一个单独的管理网络（可选）

❑ 安装所需的数据。

6.3. 为 OPENSHIFT 安装设置环境

6.3.1. 在置备程序节点上安装 RHEL

在网络配置完成后，下一步是在置备程序节点上安装 RHEL 8.x。在安装 OpenShift Container Platform 集群时，安装程序使用 provisioner 节点作为编配器。在本文档中，在置备程序节点上安装 RHEL 超出了范围。但是，选项包括但不局限于使用 RHEL Satellite 服务器、PXE 或安装介质。

6.3.2. 为 OpenShift Container Platform 安装准备 provisioner 序节点

执行以下步骤准备环境。

流程

1. 通过 **ssh** 登录到 provisioner 节点。
2. 创建一个非 root 用户（**kni**），并为该用户提供 **sudo** 权限。

```
# useradd kni
# passwd kni
# echo "kni ALL=(root) NOPASSWD:ALL" | tee -a /etc/sudoers.d/kni
# chmod 0440 /etc/sudoers.d/kni
```

3. 为新用户生成 **ssh** 密钥。

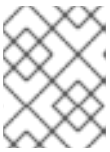
```
# su - kni -c "ssh-keygen -t ed25519 -f /home/kni/.ssh/id_rsa -N ''"
```

4. 使用新创建的用户登陆到 provisioner 节点。

```
# su - kni
$
```

5. 使用 Red Hat Subscription Manager 注册 provisioner 节点：

```
$ sudo subscription-manager register --username=<user> --password=<pass> --auto-attach
$ sudo subscription-manager repos --enable=rhel-8-for-x86_64-appstream-rpms --
enable=rhel-8-for-x86_64-baseos-rpms
```



注意

有关 Red Hat Subscription Manager 的详情，请查看使用和配置 Red Hat Subscription Manager。

6. 安装以下软件包。

```
$ sudo dnf install -y libvirt qemu-kvm mkisofs python3-devel jq ipmitool
```

7. 修改用户以便为新创建的用户中添加 **libvirt** 组。

```
$ sudo usermod --append --groups libvirt <user>
```

8. 重启 **firewalld** 并启用 **http** 服务。

```
$ sudo systemctl start firewalld
$ sudo firewall-cmd --zone=public --add-service=http --permanent
$ sudo firewall-cmd --reload
```

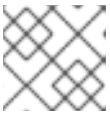
9. 启动并启用 **libvirtd** 服务。

```
$ sudo systemctl enable libvirtd --now
```

10. 创建 **default** 存储池并启动它。

```
$ sudo virsh pool-define-as --name default --type dir --target /var/lib/libvirt/images
$ sudo virsh pool-start default
$ sudo virsh pool-autostart default
```

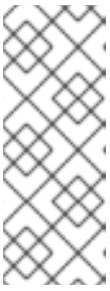
11. 配置网络。



注意

此步骤也可以从 web 控制台运行。

```
$ export PUB_CONN=<baremetal_nic_name>
$ export PROV_CONN=<prov_nic_name>
$ sudo nohup bash -c "
    nmcli con down \"$PROV_CONN\"
    nmcli con down \"$PUB_CONN\"
    nmcli con delete \"$PROV_CONN\"
    nmcli con delete \"$PUB_CONN\"
    # RHEL 8.1 appends the word \"System\" in front of the connection, delete in case it exists
    nmcli con down \"System $PUB_CONN\"
    nmcli con delete \"System $PUB_CONN\"
    nmcli connection add ifname provisioning type bridge con-name provisioning
    nmcli con add type bridge-slave ifname \"$PROV_CONN\" master provisioning
    nmcli connection add ifname baremetal type bridge con-name baremetal
    nmcli con add type bridge-slave ifname \"$PUB_CONN\" master baremetal
    pkill dhclient;dhclient baremetal
    nmcli connection modify provisioning ipv6.addresses fd00:1101::1/64 ipv6.method manual
    nmcli con down provisioning
    nmcli con up provisioning
"
```



注意

执行此步骤后 **ssh** 连接可能会断开。

只要无法通过 **baremetal** 网络路由，IPv6 地址可以是任何地址。

在使用 IPv6 地址时，请确保启用了 UEFI，并且将 UEFI PXE 设置设为 IPv6 协议。

12. 在 **provisioning** 网络连接上配置 IPv4 地址。

```
$ nmcli connection modify provisioning ipv4.addresses 172.22.0.254/24 ipv4.method manual
```

13. 重新 **ssh** 到 **provisioner** 节点（如果需要）。

```
# ssh kni@provisioner.<cluster-name>.<domain>
```

14. 验证连接桥接是否已正确创建。

```
$ sudo nmcli con show
```

```
NAME               UUID                                  TYPE      DEVICE
baremetal          4d5133a5-8351-4bb9-bfd4-3af264801530  bridge    baremetal
provisioning       43942805-017f-4d7d-a2c2-7cb3324482ed  bridge    provisioning
virbr0             d9bca40f-eee1-410b-8879-a2d4bb0465e7  bridge    virbr0
bridge-slave-eno1  76a8ed50-c7e5-4999-b4f6-6d9014dd0812  ethernet  eno1
bridge-slave-eno2  f31c3353-54b7-48de-893a-02d2b34c4736  ethernet  eno2
```

15. 创建一个 **pull-secret.txt** 文件。

```
$ vim pull-secret.txt
```

在 Web 浏览器中，导航到 Install OpenShift on Bare Metal with Installer-provisioned infrastructure，然后滚动到 **Downloads** 部分。点 **Copy pull secret**。将内容粘贴到 **pull-secret.txt** 文件中，并将内容保存到 **kni** 用户的主目录中。

6.3.3. 检索 OpenShift Container Platform 安装程序

使用安装程序的 **latest-4.x** 版本来部署最新的 OpenShift Container Platform 发行版本：

```
$ export VERSION=latest-4.6
export RELEASE_IMAGE=$(curl -s https://mirror.openshift.com/pub/openshift-v4/clients/ocp/$VERSION/release.txt | grep 'Pull From: quay.io' | awk -F ' ' '{print $3}')
```

其他资源

- 如需了解不同 发行频道的信息，请参阅 OpenShift Container Platform 升级 频道和发行版本。

6.3.4. 提取 OpenShift Container Platform 安装程序

在获取安装程序后，下一步就是展开它。

流程

1. 设置环境变量：

```
$ export cmd=openshift-baremetal-install
$ export pullsecret_file=~/pull-secret.txt
$ export extract_dir=$(pwd)
```

2. 获取 **oc** 二进制文件：

```
$ curl -s https://mirror.openshift.com/pub/openshift-v4/clients/ocp/$VERSION/openshift-client-linux.tar.gz | tar zxvf - oc
```

3. 展开安装程序：

```
$ sudo cp oc /usr/local/bin
$ oc adm release extract --registry-config "${pullsecret_file}" --command=$cmd --to "${extract_dir}" ${RELEASE_IMAGE}
$ sudo cp openshift-baremetal-install /usr/local/bin
```

6.3.5. 创建 RHCOS 镜像缓存（可选）

要使用镜像缓存，您必须下载两个镜像：Bootstrap 虚拟机使用的 Red Hat Enterprise Linux CoreOS（RHCOS）镜像，以及安装程序用来置备不同节点的 RHCOS 镜像。镜像缓存是可选的，但在有限带宽的网络中运行安装程序时特别有用。

如果您在带有有限带宽的网络上运行安装程序，而 RHCOS 镜像下载时间超过 15 到 20 分钟，则安装程序会超时。在这样的情况下，可以将镜像缓存到网页服务器上。

使用以下步骤安装包含镜像的容器。

1. 安装 **podman**。

```
$ sudo dnf install -y podman
```

2. 打开防火墙端口 **8080** 以用于 RHCOS 镜像缓存。

```
$ sudo firewall-cmd --add-port=8080/tcp --zone=public --permanent
```

```
$ sudo firewall-cmd --reload
```

3. 创建用于存储 **bootstraposimage** 和 **clusterosimage** 的目录。

```
$ mkdir /home/kni/rhcos_image_cache
```

4. 为新创建的目录设置正确的 SELinux 上下文。

```
$ sudo semanage fcontext -a -t httpd_sys_content_t "/home/kni/rhcos_image_cache(/.*)?"
$ sudo restorecon -Rv rhcos_image_cache/
```

5. 从安装程序获取提交 ID。这个 ID 决定了安装程序需要下载的镜像。

```
$ export COMMIT_ID=$(/usr/local/bin/openshift-baremetal-install version | grep '^built from commit' | awk '{print $4}')
```

6. 获取安装程序将部署到节点上的 RHCOS 镜像的 URI。

```
$ export RHCOS_OPENSTACK_URI=$(curl -s -S https://raw.githubusercontent.com/openshift/installer/$COMMIT_ID/data/data/rhcos.json  | jq .images.openstack.path | sed 's/"//g')
```

7. 获取安装程序要在 bootstrap 虚拟机上部署的 RHCOS 镜像的 URI。

```
$ export RHCOS_QEMU_URI=$(curl -s -S
https://raw.githubusercontent.com/openshift/installer/$COMMIT_ID/data/data/rhcos.json  | jq
.images.qemu.path | sed 's/"//g')
```

8. 获取发布镜像的路径。

```
$ export RHCOS_PATH=$(curl -s -S
https://raw.githubusercontent.com/openshift/installer/$COMMIT_ID/data/data/rhcos.json | jq
.baseURI | sed 's/"//g')
```

9. 获取要在 bootstrap 虚拟机上部署的 RHCOS 镜像的 SHA 哈希。

```
$ export RHCOS_QEMU_SHA_UNCOMPRESSED=$(curl -s -S
https://raw.githubusercontent.com/openshift/installer/$COMMIT_ID/data/data/rhcos.json  | jq
-r '.images.qemu["uncompressed-sha256"]')
```

10. 获取要在节点上部署的 RHCOS 镜像的 SHA 哈希。

```
$ export RHCOS_OPENSTACK_SHA_COMPRESSED=$(curl -s -S
https://raw.githubusercontent.com/openshift/installer/$COMMIT_ID/data/data/rhcos.json  | jq
-r '.images.openstack.sha256')
```

11. 下载这些镜像并将其放在 **/home/kni/rhcos_image_cache** 目录中。

```
$ curl -L ${RHCOS_PATH}${RHCOS_QEMU_URI} -o
/home/kni/rhcos_image_cache/${RHCOS_QEMU_URI}
$ curl -L ${RHCOS_PATH}${RHCOS_OPENSTACK_URI} -o
/home/kni/rhcos_image_cache/${RHCOS_OPENSTACK_URI}
```

12. 对于新创建的文件，确认 SELinux 类型为 **httpd_sys_content_t**。

```
$ ls -Z /home/kni/rhcos_image_cache
```

13. 创建 pod。

```
$ podman run -d --name rhcos_image_cache \
-v /home/kni/rhcos_image_cache:/var/www/html \
-p 8080:8080/tcp \
quay.io/centos7/httpd-24-centos7:latest
```

6.3.6. 配置文件

6.3.6.1. 配置 install-config.yaml 文件

install-config.yaml 文件需要一些额外的信息。大多数信息用于指导安装程序，以便安装的集群有可用硬件的足够信息以可以完全管理它们。

1. 配置 **install-config.yaml**。更改适当的变量以匹配环境，包括 **pullSecret** 和 **sshKey**。

```
apiVersion: v1
baseDomain: <domain>
metadata:
```

```
  name: <cluster-name>
networking:
  machineCIDR: <public-cidr>
  networkType: OVNKubernetes
compute:
- name: worker
  replicas: 2 (1)
controlPlane:
  name: master
  replicas: 3
  platform:
    baremetal: {}
platform:
  baremetal:
    apiVIP: <api-ip>
    ingressVIP: <wildcard-ip>
    provisioningNetworkInterface: <NIC1>
    provisioningNetworkCIDR: <CIDR>
    hosts:
      - name: openshift-master-0
        role: master
        bmc:
          address: ipmi://<out-of-band-ip> (2)
          username: <user>
          password: <password>
        bootMACAddress: <NIC1-mac-address>
        hardwareProfile: default
      - name: <openshift-master-1>
        role: master
        bmc:
          address: ipmi://<out-of-band-ip> (3)
          username: <user>
          password: <password>
        bootMACAddress: <NIC1-mac-address>
        hardwareProfile: default
      - name: <openshift-master-2>
        role: master
        bmc:
          address: ipmi://<out-of-band-ip> (4)
          username: <user>
          password: <password>
        bootMACAddress: <NIC1-mac-address>
        hardwareProfile: default
      - name: <openshift-worker-0>
        role: worker
        bmc:
          address: ipmi://<out-of-band-ip> (5)
          username: <user>
          password: <password>
        bootMACAddress: <NIC1-mac-address>
        hardwareProfile: unknown
      - name: <openshift-worker-1>
        role: worker
        bmc:
          address: ipmi://<out-of-band-ip>
```

```
        username: <user>
        password: <password>
      bootMACAddress: <NIC1-mac-address>
      hardwareProfile: unknown
pullSecret: '<pull_secret>'
sshKey: '<ssh_pub_key>'
```

1 根据作为 OpenShift Container Platform 集群一部分的 worker 节点数量来缩放 worker 机器。

2 3 4 5 如需了解更多选项，请参阅 BMC 寻址部分。

2. 创建用于存储集群配置的目录。

```
$ mkdir ~/clusterconfigs
$ cp install-config.yaml ~/clusterconfigs
```

3. 在安装 OpenShift Container Platform 集群前，请确保关闭所有裸机节点。

```
$ ipmitool -I lanplus -U <user> -P <password> -H <management-server-ip> power off
```

4. 如果以前的部署尝试中保留任何旧的 bootstrap 资源，则删除旧的 bootstrap 资源。

```
for i in $(sudo virsh list | tail -n +3 | grep bootstrap | awk {'print $2'});
do
  sudo virsh destroy $i;
  sudo virsh undefine $i;
  sudo virsh vol-delete $i --pool $i;
  sudo virsh vol-delete $i.ign --pool $i;
  sudo virsh pool-destroy $i;
  sudo virsh pool-undefine $i;
done
```

6.3.6.2. 在 install-config.yaml 文件中设置代理设置（可选）

要使用代理部署 OpenShift Container Platform 集群，请对 **install-config.yaml** 文件进行以下更改。

```
apiVersion: v1
baseDomain: <domain>
proxy:
  httpProxy: http://USERNAME:PASSWORD@proxy.example.com:PORT
  httpsProxy: https://USERNAME:PASSWORD@proxy.example.com:PORT
  noProxy: <WILDCARD_OF_DOMAIN>,<PROVISIONING_NETWORK/CIDR>,
<BMC_ADDRESS_RANGE/CIDR>
```

以下是带有值的 **noProxy** 示例。

```
noProxy: .example.com,172.22.0.0/24,10.10.0.0/24
```

启用代理后，请在对应的键/值对中设置代理的适当值。

主要考虑：

- 如果代理没有 HTTPS 代理，将 **httpsProxy** 的值从 **https://** 改为 **http://**。
- 如果使用 provisioning 网络，将其包含在 **noProxy** 设置中，否则安装程序将失败。
- 将所有代理设置设置为 provisioner 节点中的环境变量。例如：**HTTP_PROXY**、**HTTPS_PROXY** 和 **NO_PROXY**。

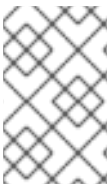
注意

使用 IPv6 置备时，您无法在 **noProxy** 设置中定义 CIDR 地址块。您必须单独定义每个地址。

6.3.6.3. 针对没有 provisioning 的网络修改 install-config.yaml 文件（可选）

要在没有 **provisioning** 网络的情况下部署 OpenShift Container Platform 集群，需要对 **install-config.yaml** 文件进行以下更改。

```
platform:
  baremetal:
    apiVIP: <apiVIP>
    ingressVIP: <ingress/wildcard VIP>
    provisioningNetwork: "Disabled"
    provisioningHostIP: <baremetal_network_IP1>
    bootstrapProvisioningIP: <baremetal_network_IP2>
```



注意

对于 **provisioningHostIP** 和 **bootstrapProvisioningIP** 配置设置，需要提供两个来自 **baremetal** 网络的 IP 地址，并删除 **provisioningBridge** 和 **provisioningNetworkCIDR** 配置设置。

6.3.6.4. 额外的 install-config 参数

以下表格中包括了 **install-config.yaml** 文件所需的参数、**hosts** 参数和 **bmc** 参数。

表 6.1. 所需的参数

参数	默认	描述
baseDomain		集群的域名。例如：**example.com**。
sshKey		**sshKey** 配置设置包含 **~/.ssh/id_rsa.pub** 文件中访问 control plane 节点和 worker 节点所需的密钥。通常，这个密钥来自 **provisioner** 节点。
pullSecret		**pullSecret** 配置设置包含准备置备程序节点时从 Install OpenShift on Bare Metal 页中下载的 pull secret 的副本。

参数	默认	描述
metadata: name:		给 OpenShift Container Platform 集群的名称。例如, **openshift**。
networking: machineCIDR:		外部网络的公共 CIDR（Classless Inter-Domain Routing）。例如：**10.0.0.0/24** .
compute: - name: worker		OpenShift Container Platform 集群需要为 worker（或 compute）节点提供名称，即使没有节点也是如此。
compute: replicas: 2		Replicas 设置 OpenShift Container Platform 集群中的 worker（或 compute）节点的数量。
controlPlane: name: master		OpenShift Container Platform 集群需要一个 control plane（master）节点的名称。
controlPlane: replicas: 3		replicas 设置作为 OpenShift Container Platform 集群一部分的 control plane（master）节点的数量。
provisioningNetworkInterface		连接到 provisioning 网络的 control plane 节点上的网络接口名称。
defaultMachinePlatform		用于没有平台配置的机器池的默认配置。
apiVIP	**api.<clustername.clusterdomain>**	用于内部 API 通信的 VIP。 这个设置必须提供，或者在 DNS 中预先配置，以便正确解析默认名称。
disableCertificateVerification	**False**	**redfish** 和 **redfish-virtualmedia** 需要这个参数来管理 BMC 地址。当 BMC 地址使用自签名证书时，这个值应该是 **True**。
ingressVIP	**test.apps.<clustername.clusterdomain>**	用于入口流量的 VIP。

表 6.2. 可选参数

参数	默认	描述
provisioningDHCPRange	**172.22.0.10,172.22.0.100**	定义 **provisioning** 网络上节点的 IP 范围。
provisioningNetworkCIDR	**172.22.0.0/24**	用于置备的网络的 CIDR。在 **provisioning** 网络中不使用默认地址范围时需要这个选项。
clusterProvisioningIP	**provisioningNetworkCIDR** 的第三个 IP 地址。	运行置备服务的集群中的 IP 地址。默认为 **provisioning** 子网的第三个 IP 地址。例如： **172.22.0.3**。
bootstrapProvisioningIP	**provisioningNetworkCIDR** 的第二个 IP 地址。	在安装程序部署 control plane（master）节点时运行置备服务的 bootstrap 虚拟机上的 IP。默认为 **provisioning** 子网的第二个 IP。例如, **172.22.0.2**。 在没有 **provisioning** 网络时，将此值设置为 **baremetal** 网络中的一个可用的 IP 地址。
externalBridge	**baremetal**	附加到 **baremetal** 网络的 hypervisor**baremetal** 网桥的名称。
provisioningBridge	**provisioning**	附加到 **provisioning** 网络的 **provisioner** 主机上的 **provisioning** 网桥的名称。
defaultMachinePlatform		用于没有平台配置的机器池的默认配置。
bootstrapOSImage		用于覆盖 bootstrap 节点的默认操作系统镜像的 URL。URL 必须包含镜像的 SHA-256 哈希。例如：**https://mirror.openshift.com/rhcos-<version>-qemu.qcow2.gz?sha256=<uncompressed_sha256>**;。
clusterOSImage		用于覆盖集群节点默认操作系统的 URL。URL 必须包含镜像的 SHA-256 哈希。例如，**https://mirror.openshift.com/images/rhcos-<version>-openstack.qcow2.gz?sha256=<compressed_sha256>**;。
provisioningNetwork		将这个参数设置为 **Disabled** 以禁用 **provisioning** 网络的要求。用户只能执行基于虚拟介质的置备，或使用协助的安装使集群启用。如果使用电源管理，则需要从机器网络访问 BMC。用户必须在外部网络上提供两个用于置备服务的 IP 地址。将此参数设置为 **managed**（默认参数）来完全管理 provisioning 网络，包括 DHCP、TFTP 等等。 将此参数设置为 **非受管状态**，仍然可启用置备网络，但需要手动配置 DHCP。建议使用虚拟介质置备，但在需要时仍可使用 PXE。

参数	默认	描述
provisioningHostingIp		当 **provisioningNetwork** 配置设置为 **Disabled** 时，把这个参数设为**baremetal** 网络中的一个可用的 IP 地址。
httpProxy		将此参数设置为环境中使用的适当 HTTP 代理。
httpsProxy		将此参数设置为环境中使用的适当 HTTPS 代理。
noProxy		将这个参数设置为适合环境中代理使用的排除项。

Hosts

hosts 参数是用于构建集群的独立裸机资产列表。

名称	默认	描述
名称		与详情关联的 **BareMetalHost** 资源的名称。例如, **openshift-master-0**。
role		裸机节点的角色。**master** 或 **worker**。
bmc		基板管理控制器的连接详情。如需了解更多详细信息，请参阅 BMC 寻址部分。
bootMACAddress		主机用来在 **provisioning** 网络中引导的 NIC 的 MAC 地址。

6.3.6.5. BMC 地址

每个 **bmc** 条目的 **address** 字段都是连接到 OpenShift Container Platform 集群节点的 URL，包括 URL 方案中的控制器类型以及在网络中的位置。

IPMI

IPMI 主机使用 **ipmi://<out-of-band-ip>:<port>**，如果没有指定，默认使用端口 **623**。以下示例演示了 **install-config.yaml** 文件中的 IPMI 配置。

```
platform:
  baremetal:
    hosts:
      - name: openshift-master-0
        role: master
        bmc:
```

```
      address: ipmi://<out-of-band-ip>
      username: <user>
      password: <password>
```

RedFish for HPE

要启用 RedFish，使用 **redfish://** 或 **redfish+http://** 禁用 TLS。安装程序需要主机名或者 IP 地址以及到系统 ID 的路径。以下示例演示了 **install-config.yaml** 文件中的 RedFish 配置。

```
platform:
  baremetal:
    hosts:
      - name: openshift-master-0
        role: master
        bmc:
          address: redfish://<out-of-band-ip>/redfish/v1/Systems/1
          username: <user>
          password: <password>
```

虽然建议为带外管理地址提供颁发机构证书，但在使用自签名证书时，您必须在 **bmc** 配置中包括 **disableCertificateVerification: True**。以下示例演示了在 **install-config.yaml** 文件中使用 **disableCertificateVerification: True** 配置参数的 RedFish 配置。

```
platform:
  baremetal:
    hosts:
      - name: openshift-master-0
        role: master
        bmc:
          address: redfish://<out-of-band-ip>/redfish/v1/Systems/1
          username: <user>
          password: <password>
          disableCertificateVerification: True
```

RedFish for Dell

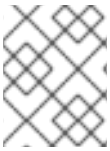
要启用 RedFish，使用 **redfish://** 或 **redfish+http://** 禁用 TLS。安装程序需要主机名或者 IP 地址以及到系统 ID 的路径。以下示例演示了 **install-config.yaml** 文件中的 RedFish 配置。

```
platform:
  baremetal:
    hosts:
      - name: openshift-master-0
        role: master
        bmc:
          address: redfish://<out-of-band-ip>/redfish/v1/Systems/System.Embedded.1
          username: <user>
          password: <password>
```

虽然建议为带外管理地址提供颁发机构证书，但在使用自签名证书时，您必须在 **bmc** 配置中包括 **disableCertificateVerification: True**。以下示例演示了在 **install-config.yaml** 文件中使用 **disableCertificateVerification: True** 配置参数的 RedFish 配置。

```
platform:
  baremetal:
```

```
  hosts:
    - name: openshift-master-0
      role: master
      bmc:
        address: redfish://<out-of-band-ip>/redfish/v1/Systems/System.Embedded.1
        username: <user>
        password: <password>
        disableCertificateVerification: True
```



注意

目前，只有带有 iDRAC 固件版本 **4.20.20.20** 或更高版本的 Dell 支持 RedFish，用于裸机部署上的安装程序置备的 OpenShift Container Platform 安装。

RedFish Virtual Media for HPE

要为 HPE 服务器启用 RedFish Virtual Media，在 **address** 设置中使用 **redfish-virtualmedia://**。以下示例演示了在 **install-config.yaml** 文件中使用 RedFish Virtual Media。

```
platform:
  baremetal:
    hosts:
      - name: openshift-master-0
        role: master
        bmc:
          address: redfish-virtualmedia://<out-of-band-ip>/redfish/v1/Systems/1
          username: <user>
          password: <password>
```

RedFish Virtual Media for Dell

对于 Dell 服务器中的 RedFish Virtual Media，在 **address** 设置中使用 **idrac-virtualmedia://**。

注意

Dell 服务器中的 redfish Virtual Media 在 OpenShift Container Platform 4.6 中有一个已知的问题。4.6.1 点发行版本会解决这个问题。

以下示例演示了在 **install-config.yaml** 文件中使用 iDRAC Virtual Media。

```
platform:
  baremetal:
    hosts:
      - name: openshift-master-0
        role: master
        bmc:
          address: idrac-virtualmedia://<out-of-band-ip>/redfish/v1/Systems/System.Embedded.1
          username: <user>
          password: <password>
```

注意

idrac-virtualmedia 需要 iDRAC 固件版本 4.20.20.20 或更高版本。

通过 iDRAC 控制台，确保 OpenShift Container Platform 集群节点带有 AutoAttach Enabled。菜单路径是：**Configuration→Virtual Media→Attach Mode→AutoAttach**.

6.3.6.6. Root device hints

rootDeviceHints 参数使安装程序能够将 Red Hat Enterprise Linux CoreOS（RHCOS）镜像置备到特定设备。安装程序会按照发现设备的顺序检查设备，并将发现的值与 hint 值进行比较。安装程序会使用第一个与 hint 值匹配的发现设备。该配置可包括多个 hint，但只有与所有 hint 都匹配的设备才会被安装程序选择。

表 6.3. 子字段

子字段	描述
deviceName	包含类似 **/dev/vda** 的 Linux 设备名称的字符串。hint 必须与实际值完全匹配。
hctl	包含类似 **0:0:0:0** 的 SCSI 总线地址的字符串。hint 必须与实际值完全匹配。
model	包含特定厂商的设备标识符的字符串。hint 可以是实际值的子字符串。
vendor	包含该设备厂商或制造商名称的字符串。hint 可以是实际值的子字符串。
serialNumber	包含设备序列号的字符串。hint 必须与实际值完全匹配。
minSizeGigabytes	代表设备最小值的一个整数，以 GB 为单位。
wwn	包含唯一存储标识符的字符串。hint 必须与实际值完全匹配。
wwnWithExtension	包含唯一存储标识符且附加厂商扩展的字符串。hint 必须与实际值完全匹配。
wwnVendorExtension	包含唯一厂商存储标识符的字符串。hint 必须与实际值完全匹配。
rotational	指明该设备为旋转磁盘（true）还是非旋转磁盘（false）的布尔值。

示例用法

```
- name: master-0
  role: master
```

```
    bmc:
      address: ipmi://10.10.0.3:6203
      username: admin
      password: redhat
    bootMACAddress: de:ad:be:ef:00:40
    rootDeviceHints:
      deviceName: "/dev/sda"
```

6.3.6.7. 创建 OpenShift Container Platform 清单

1. 创建 OpenShift Container Platform 清单。

```
$ ./openshift-baremetal-install --dir ~/clusterconfigs create manifests
```

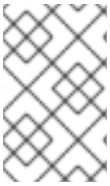
```
INFO Consuming Install Config from target directory
WARNING Making control-plane schedulable by setting MastersSchedulable to true for
Scheduler cluster settings
WARNING Discarding the OpenShift Manifest that was provided in the target directory
because its dependencies are dirty and it needs to be regenerated
```

6.3.7. 创建断开连接的 registry（可选）

在某些情况下,您可能想要使用安装 registry 的本地副本安装 OpenShift KNI 集群。这可能会提高网络效率，因为集群节点位于无法访问互联网的网络中。

一个本地镜像的 registry 副本需要以下内容：

- registry 节点的证书。这可以是自签名证书。
- 一个 webserver - 这由系统中的容器提供。
- 一个更新的 pull secret，其中包含证书和本地存储库信息。



注意

在 registry 节点上创建断开连接的 registry 是可选的。在后续的小节中可以看到它们是可选的，只有在 registry 节点上创建断开连接的 registry 时才需要执行相关的步骤。在 registry 节点上创建断开连接的 registry 时，您应该执行标记为"（可选）" 的子章节。

6.3.7.1. 准备 registry 节点以托管已镜像的 registry（可选）

对 registry 节点进行以下更改。

流程

1. 打开 registry 节点上的防火墙端口。

```
$ sudo firewall-cmd --add-port=5000/tcp --zone=libvirt  --permanent
$ sudo firewall-cmd --add-port=5000/tcp --zone=public   --permanent
$ sudo firewall-cmd --reload
```

2. 为 registry 节点安装所需的软件包。

```
$ sudo yum -y install python3 podman httpd httpd-tools jq
```

3. 创建保存存储库信息的目录结构。

```
$ sudo mkdir -p /opt/registry/{auth,certs,data}
```

6.3.7.2. 生成自签名证书（可选）

为 registry 节点生成自签名证书，并将其放在 **/opt/registry/certs** 目录中。

流程

1. 根据情况调整证书信息。

```
$ host_fqdn=$( hostname --long )
$ cert_c="<Country Name>"   # Country Name (C, 2 letter code)
$ cert_s="<State>"          # Certificate State (S)
$ cert_l="<Locality>"       # Certificate Locality (L)
$ cert_o="<Organization>"   # Certificate Organization (O)
$ cert_ou="<Org Unit>"      # Certificate Organizational Unit (OU)
$ cert_cn="${host_fqdn}"    # Certificate Common Name (CN)

$ openssl req \
    -newkey rsa:4096 \
    -nodes \
    -sha256 \
    -keyout /opt/registry/certs/domain.key \
    -x509 \
    -days 365 \
    -out /opt/registry/certs/domain.crt \
    -addext "subjectAltName = DNS:${host_fqdn}" \
    -subj "/C=${cert_c}/ST=${cert_s}/L=${cert_l}/O=${cert_o}/OU=${cert_ou}/CN=${cert_cn}"
```

注意

当替换 **<Country Name>** 时，请确保它只包含两个字母。例如，**US**。

2. 使用新证书更新 registry 节点的 **ca-trust**。

```
$ sudo cp /opt/registry/certs/domain.crt /etc/pki/ca-trust/source/anchors/
$ sudo update-ca-trust extract
```

6.3.7.3. 创建 registry podman 容器（可选）

registry 容器使用 **/opt/registry** 目录来保存证书、认证文件以及存储其数据文件。

registry 容器使用 **httpd**，并需要 **htpasswd** 文件进行身份验证。

流程

1. 在 **/opt/registry/auth** 中创建一个 **htpasswd** 文件供容器使用。

```
$ htpasswd -bBc /opt/registry/auth/htpasswd <user> <passwd>
```

将 **<user>** 替换为用户名，将 **<passwd>** 替换为密码。

2. 创建并启动 registry 容器。

```
$ podman create \
  --name ocpdiscon-registry \
  -p 5000:5000 \
  -e "REGISTRY_AUTH=htpasswd" \
  -e "REGISTRY_AUTH_HTPASSWD_REALM=Registry" \
  -e "REGISTRY_HTTP_SECRET=ALongRandomSecretForRegistry" \
  -e "REGISTRY_AUTH_HTPASSWD_PATH=/auth/htpasswd" \
  -e "REGISTRY_HTTP_TLS_CERTIFICATE=/certs/domain.crt" \
  -e "REGISTRY_HTTP_TLS_KEY=/certs/domain.key" \
  -e "REGISTRY_COMPATIBILITY_SCHEMA1_ENABLED=true" \
  -v /opt/registry/data:/var/lib/registry:z \
  -v /opt/registry/auth:/auth:z \
  -v /opt/registry/certs:/certs:z \
  docker.io/library/registry:2
```

```
$ podman start ocpdiscon-registry
```

6.3.7.4. 复制和更新 pull-secret（可选）

将 pull secret 文件从置备程序节点复制到 registry 节点，并修改该文件，使其包含新 registry 节点的身份验证信息。

流程

1. 复制 **pull-secret.txt** 文件。

```
$ scp kni@provisioner:/home/kni/pull-secret.txt pull-secret.txt
```

2. 使用 registry 节点的完全限定域名更新 **host_fqdn** 环境变量。

```
$ host_fqdn=$( hostname --long )
```

3. 使用用于创建 **htpasswd** 文件的 **http** 凭证的 base64 编码来更新 **b64auth** 环境变量。

```
$ b64auth=$( echo -n '<username>:<passwd>' | openssl base64 )
```

将 **<username>** 替换为用户名，将 **<passwd>** 替换为密码。

4. 设置 **AUTHSTRING** 环境变量使用 **base64** 授权字符串。**$USER** 变量是包含当前用户名称的环境变量。

```
$ AUTHSTRING="{\"$host_fqdn:5000\": {\"auth\": \"$b64auth\",\"email\":
\"$USER@redhat.com\"}}"
```

5. 更新 **pull-secret.txt** 文件。

```
$ jq ".auths += $AUTHSTRING" < pull-secret.txt > pull-secret-update.txt
```

6.3.7.5. 对存储库进行镜像（可选）

流程

1. 将 provisioner 节点中的 **oc** 二进制文件复制到 registry 节点。

```
$ sudo scp kni@provisioner:/usr/local/bin/oc /usr/local/bin
```

2. 设置所需的环境变量。
   a. 设置发行版本：

```
$ VERSION=<release_version>
```

   对于 **<release_version>**，请指定与 OpenShift Container Platform 版本对应的标签，如 **4.6**。

   b. 设置本地 registry 名称和主机端口：

```
$ LOCAL_REG='<local_registry_host_name>:<local_registry_host_port>'
```

   对于 **<local_registry_host_name>**，请指定镜像存储库的 registry 域名；对于 **<local_registry_host_port>**，请指定用于提供内容的端口。

   c. 设置本地存储库名称：

```
$ LOCAL_REPO='<local_repository_name>'
```

   对于 **<local_repository_name>**，请指定要在 registry 中创建的仓库名称，如 **ocp4/openshift4**。

3. 将远程安装镜像镜像(mirror)到本地存储库。

```
$ /usr/local/bin/oc adm release mirror \
  -a pull-secret-update.txt \
  --from=$UPSTREAM_REPO \
  --to-release-image=$LOCAL_REG/$LOCAL_REPO:${VERSION} \
  --to=$LOCAL_REG/$LOCAL_REPO
```

6.3.7.6. 修改 install-config.yaml 文件，使用断开连接的 registry（可选）

在 provisioner 节点上，**install-config.yaml** 文件应该使用从 **pull-secret-update.txt** 文件中新创建的 pull-secret。**install-config.yaml** 文件还必须包含断开连接的 registry 节点的证书和 registry 信息。

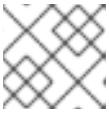
流程

1. 将断开连接的 registry 节点的证书添加到 **install-config.yaml** 文件中。证书应该跟着 **"additionalTrustBundle: |"** 行，并带有正确的缩进（通常是两个空格）。

```
$ echo "additionalTrustBundle: |" >> install-config.yaml
$ sed -e 's/^/  /' /opt/registry/certs/domain.crt >> install-config.yaml
```

2. 将 registry 的镜像信息添加到 **install-config.yaml** 文件中。

```
$ echo "imageContentSources:" >> install-config.yaml
$ echo "- mirrors:" >> install-config.yaml
$ echo "  - registry.example.com:5000/ocp4/openshift4" >> install-config.yaml
$ echo "  source: quay.io/openshift-release-dev/ocp-release" >> install-config.yaml
$ echo "- mirrors:" >> install-config.yaml
$ echo "  - registry.example.com:5000/ocp4/openshift4" >> install-config.yaml
$ echo "  source: quay.io/openshift-release-dev/ocp-v4.0-art-dev" >> install-config.yaml
```



注意

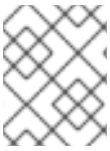
将 **registry.example.com** 替换为 registry 的完全限定域名。

6.3.8. 在 worker 节点上部署路由器

在安装过程中，安装程序会在 worker 节点上部署路由器 Pod。默认情况下，安装程序会安装两个路由器 Pod。如果初始集群只有一个 worker 节点，或者如果部署的集群需要额外的路由器来处理用于 OpenShift Container Platform 集群中服务的外部流量负载，您可以创建一个 **yaml** 文件来设置适当数量的路由器副本。

注意

默认情况下，安装程序会部署两个路由器。如果集群至少有两个 worker 节点，您可以跳过此部分。如需有关 Ingress Operator 的更多信息，请参阅 OpenShift Container Platform 中的 Ingress Operator。



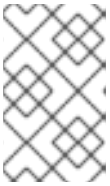
注意

如果集群没有 worker 节点，安装程序默认会在 control plane 节点上部署两个路由器。如果集群没有 worker 节点，可以跳过本节。

流程

1. 创建一个 **router-replicas.yaml** 文件。

```
apiVersion: operator.openshift.io/v1
kind: IngressController
metadata:
  name: default
  namespace: openshift-ingress-operator
spec:
  replicas: <num-of-router-pods>
  endpointPublishingStrategy:
    type: HostNetwork
  nodePlacement:
    nodeSelector:
      matchLabels:
        node-role.kubernetes.io/worker: ""
```



注意

将 **<num-of-router-pods>** 替换为适当的值。如果只使用一个 worker 节点，将 **replicas:** 设置为 **1**。如果使用 3 个以上 worker 节点，您可以根据情况增加 **replicas:** 的默认值（**2**）。

2. 将 **router-replicas.yaml** 文件保存并复制到 **clusterconfigs/openshift** 目录中。

```
cp ~/router-replicas.yaml clusterconfigs/openshift/99_router-replicas.yaml
```

6.3.9. 安装的验证清单

❑ 已检索到 OpenShift Container Platform 安装程序。

❑ 已展开 OpenShift Container Platform 安装程序。

❑ 已配置了 **install-config.yaml** 所需的参数。

❑ 已为 **install-config.yaml** 配置了 **hosts** 参数。

❑ 已配置 **install-config.yaml** 的 **bmc** 参数。

❑ 在 **bmc** 的 **address** 字段中配置的值已被应用。

❑ 创建断开连接的 registry（可选）。

❑ （可选）如果使用，断开连接的 registry 设置。

❑ （可选）在 worker 节点上部署了路由。

6.3.10. 通过 OpenShift Container Platform 安装程序部署集群

运行 OpenShift Container Platform 安装程序：

```
$ ./openshift-baremetal-install --dir ~/clusterconfigs --log-level debug create cluster
```

6.3.11. 安装后

在部署过程中，您可以通过对安装目录文件夹中的 **.openshift_install.log** 日志文件发出 **tail** 命令来检查安装的整体状态。

```
$ tail -f /path/to/install-dir/.openshift_install.log
```

6.3.12. 准备在裸机上重新安装集群

在在裸机上重新安装集群前，您必须执行清理操作。

流程

1. 删除或重新格式化 bootstrap、control plane（也称为 master）节点和 worker 节点的磁盘。如果您在虚拟机监控程序环境中工作，您必须添加您要删除的任何磁盘。

2. 删除之前生成的工件：

```
$ cd ; /bin/rm -rf auth/ bootstrap.ign master.ign worker.ign metadata.json \
.openshift_install.log .openshift_install_state.json
```

3. 生成新清单和 Ignition 配置文件。如需更多信息，请参阅"创建 Kubernetes 清单和 Ignition 配置文件"。

4. 将安装程序创建的新 bootstrap、control plane 和计算节点 Ignition 配置文件上传到 HTTP 服务器。这将覆盖以前的 Ignition 文件。

6.4. 扩展集群

部署安装程序置备的 OpenShift Container Platform 集群后，您可以使用以下步骤扩展 worker 节点的数量。确保每个 worker 节点都满足先决条件。

6.4.1. 准备裸机节点

准备裸机节点需要从 provisioner 节点执行以下步骤。

流程

1. 如果需要，获取 **oc** 二进制文件。它应该已存在于 provisioner 节点上。

```
$ curl -s https://mirror.openshift.com/pub/openshift-v4/clients/ocp-dev-
preview/$VERSION/openshift-client-linux.tar.gz | tar zxvf - oc
```

```
$ sudo cp oc /usr/local/bin
```

2. 安装 **ipmitool**。

```
$ sudo dnf install -y OpenIPMI ipmitool
```

3. 关闭裸机节点并确保它已关闭。

```
$ ipmitool -I lanplus -U <user> -P <password> -H <management-server-ip> power off
```

其中 **<management-server-ip>** 是裸机节点的基本管理控制器的 IP 地址。

```
$ ipmitool -I lanplus -U <user> -P <password> -H <management-server-ip> power status
```

```
Chassis Power is off
```

4. 检索裸机节点基板管理控制器的用户名和密码。然后，从用户名和密码创建 **base64** 字符串。在下面的示例中，用户名是 **root**，密码是 **calvin**。

```
$ echo -ne "root" | base64
```

```
$ echo -ne "calvin" | base64
```

5. 为裸机节点创建配置文件。

```
$ vim bmh.yaml
```

```
---
apiVersion: v1
kind: Secret
metadata:
  name: openshift-worker-<num>-bmc-secret
type: Opaque
data:
  username: <base64-of-uid>
  password: <base64-of-pwd>
---
apiVersion: metal3.io/v1alpha1
kind: BareMetalHost
metadata:
  name: openshift-worker-<num>
spec:
  online: true
  bootMACAddress: <NIC1-mac-address>
  bmc:
    address: ipmi://<bmc-ip>
    credentialsName: openshift-worker-<num>-bmc-secret
```

在 **name** 字段和 **credentialsName** 字段中，使用裸机节点的 worker 数量替换 **<num>**。将 **<base64-of-uid=0** 替换为用户名的 **base64** 字符串。将 **<base64-of-pwd>** 替换为密码的 **base64** 字符串。将 **<NIC1-mac-address>** 替换为裸机节点第一个 NIC 的 MAC 地址。将 **<bmc-ip>** 替换为裸机节点基板管理控制器的 IP 地址。

6. 创建裸机节点。

```
$ oc -n openshift-machine-api create -f bmh.yaml
```

```
secret/openshift-worker-<num>-bmc-secret created
baremetalhost.metal3.io/openshift-worker-<num> created
```

其中 **<num>** 是 worker 号。

7. 启动并检查裸机节点。

```
$ oc -n openshift-machine-api get bmh openshift-worker-<num>
```

其中 **<num>** 是 worker 节点号。

```
NAME                 STATUS   PROVISIONING STATUS   CONSUMER   BMC
HARDWARE PROFILE   ONLINE   ERROR
openshift-worker-<num>   OK       ready                        ipmi://<out-of-band-ip>   unknown
true
```

6.4.2. 置备裸机节点

置备裸机节点需要从 provisioner 节点执行以下步骤。

流程

1. 在置备裸机节点前，请确保 **PROVISIONING STATUS** 为 **ready**。

```
$  oc -n openshift-machine-api get bmh openshift-worker-<num>
```

其中 **<num>** 是 worker 节点号。

```
NAME                 STATUS   PROVISIONING STATUS   CONSUMER   BMC
HARDWARE PROFILE   ONLINE   ERROR
openshift-worker-<num>   OK       ready                      ipmi://<out-of-band-ip>   unknown
true
```

2. 获取 worker 节点数量。

```
$ oc get nodes
```

```
NAME                                  STATUS   ROLES          AGE     VERSION
provisioner.openshift.example.com          Ready    master         30h     v1.16.2
openshift-master-1.openshift.example.com          Ready    master         30h     v1.16.2
openshift-master-2.openshift.example.com          Ready    master         30h     v1.16.2
openshift-master-3.openshift.example.com          Ready    master         30h     v1.16.2
openshift-worker-0.openshift.example.com          Ready    master         30h     v1.16.2
openshift-worker-1.openshift.example.com          Ready    master         30h     v1.16.2
```

3. 获取机器集。

```
$ oc get machinesets -n openshift-machine-api
```

```
NAME                          DESIRED   CURRENT   READY   AVAILABLE   AGE
...
openshift-worker-0.example.com     1         1         1       1          55m
openshift-worker-1.example.com     1         1         1       1          55m
```

4. 将 worker 节点数量增加一倍。

```
$ oc scale --replicas=<num> machineset <machineset> -n openshift-machine-api
```

将 **<num>** 替换为新的 worker 节点数。将 **<machineset>** 替换为上一步中的机器集的名称。

5. 检查裸机节点的状态。

```
$ oc -n openshift-machine-api get bmh openshift-worker-<num>
```

其中 **<num>** 是 worker 节点号。状态从 **ready** 变为 **provisioning**。

```
NAME                 STATUS   PROVISIONING STATUS   CONSUMER                BMC
HARDWARE PROFILE   ONLINE   ERROR
openshift-worker-<num>   OK       provisioning           openshift-worker-<num>-65tjz
ipmi://<out-of-band-ip>   unknown          true
```

provisioning 会一直保持，直到 OpenShift Container Platform 集群置备节点。这可能需要 30 分钟或更长时间。完成后，状态将变为 **provisioned**。

```
NAME                 STATUS   PROVISIONING STATUS   CONSUMER                BMC
HARDWARE PROFILE   ONLINE   ERROR
```

```
openshift-worker-<num>   OK       provisioned          openshift-worker-<num>-65tjz
ipmi://<out-of-band-ip>  unknown  true
```

6. 置备后，请确保裸机节点就绪。

```
$ oc get nodes
```

```
NAME                                        STATUS   ROLES   AGE     VERSION
provisioner.openshift.example.com           Ready    master  30h     v1.16.2
openshift-master-1.openshift.example.com    Ready    master  30h     v1.16.2
openshift-master-2.openshift.example.com    Ready    master  30h     v1.16.2
openshift-master-3.openshift.example.com    Ready    master  30h     v1.16.2
openshift-worker-0.openshift.example.com    Ready    master  30h     v1.16.2
openshift-worker-1.openshift.example.com    Ready    master  30h     v1.16.2
openshift-worker-<num>.openshift.example.com  Ready    worker  3m27s   v1.16.2
```

您还可以检查 kubelet。

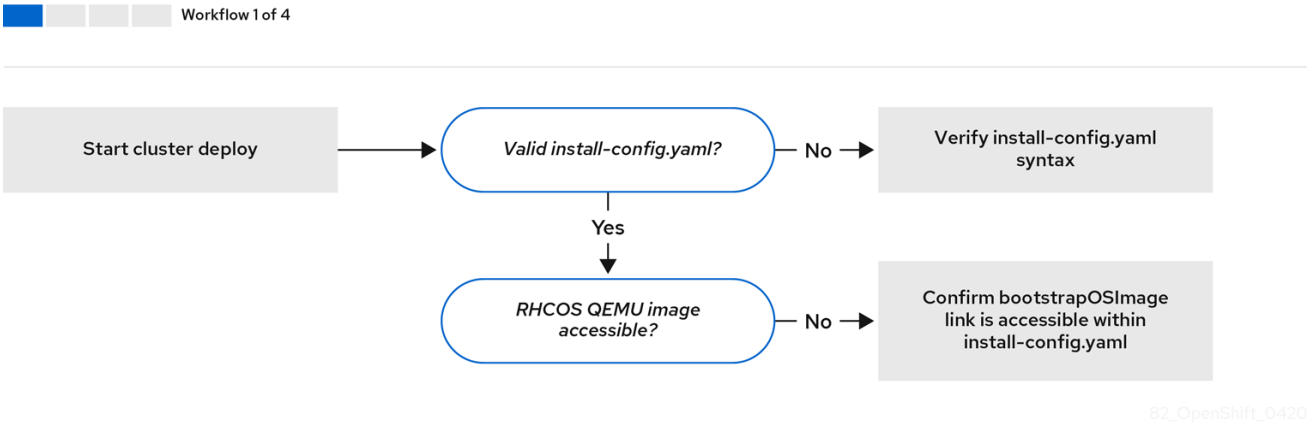
```
$ ssh openshift-worker-<num>
```

```
[kni@openshift-worker-<num>]$ journalctl -fu kubelet
```

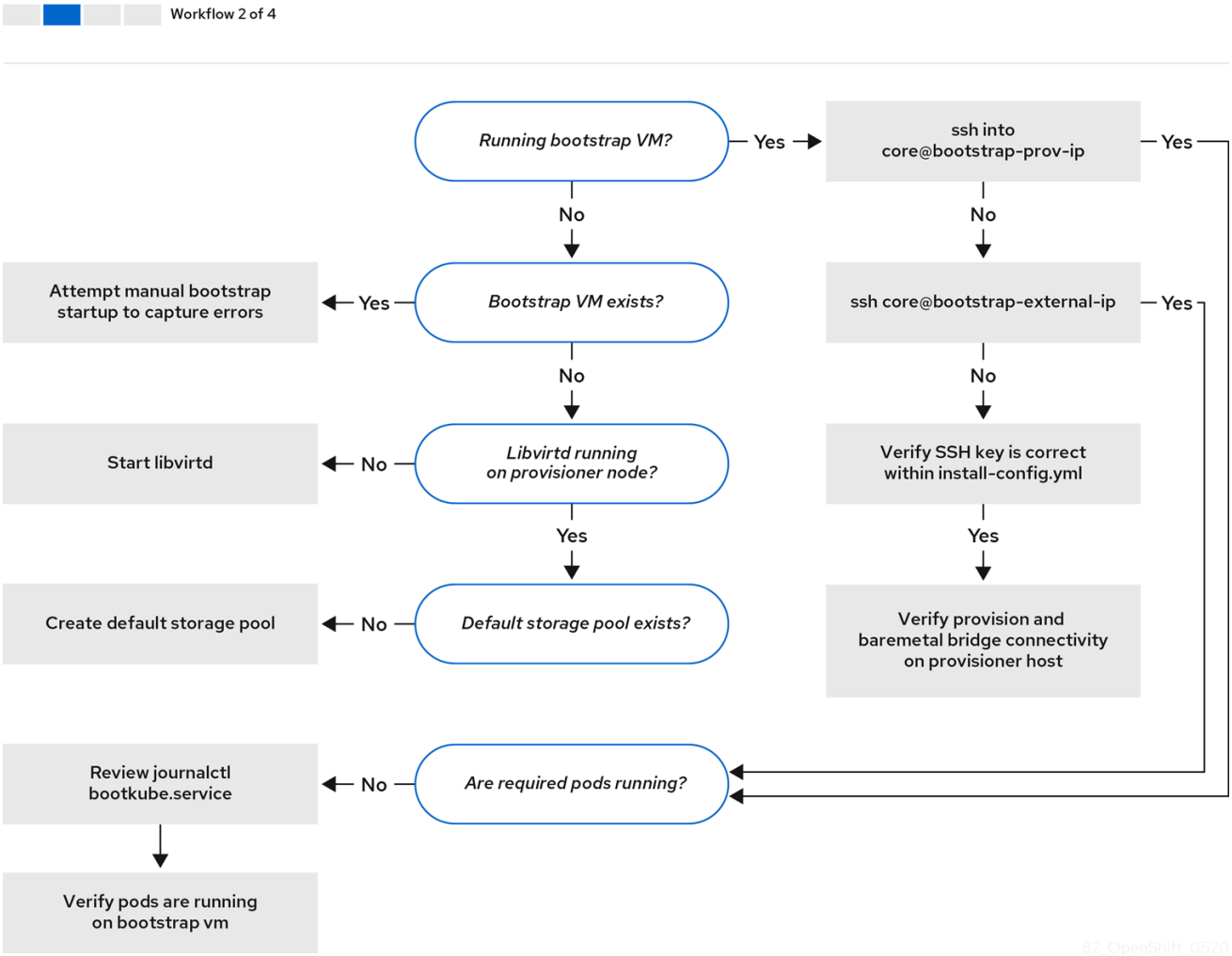
6.5. 故障排除

6.5.1. 安装程序工作流故障排除

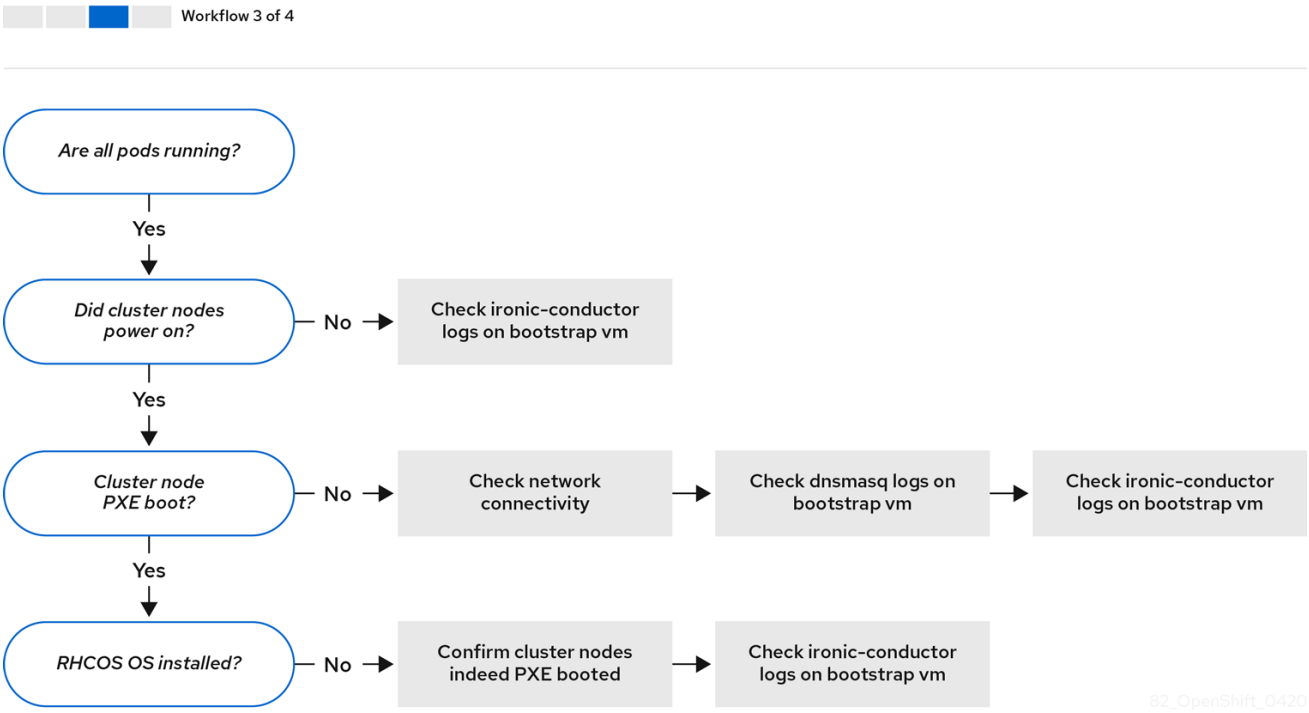
在对安装环境进行故障排除之前，了解裸机上安装程序置备安装的整体流至关重要。下面的图表提供了故障排除流程，并按部就班地划分环境。



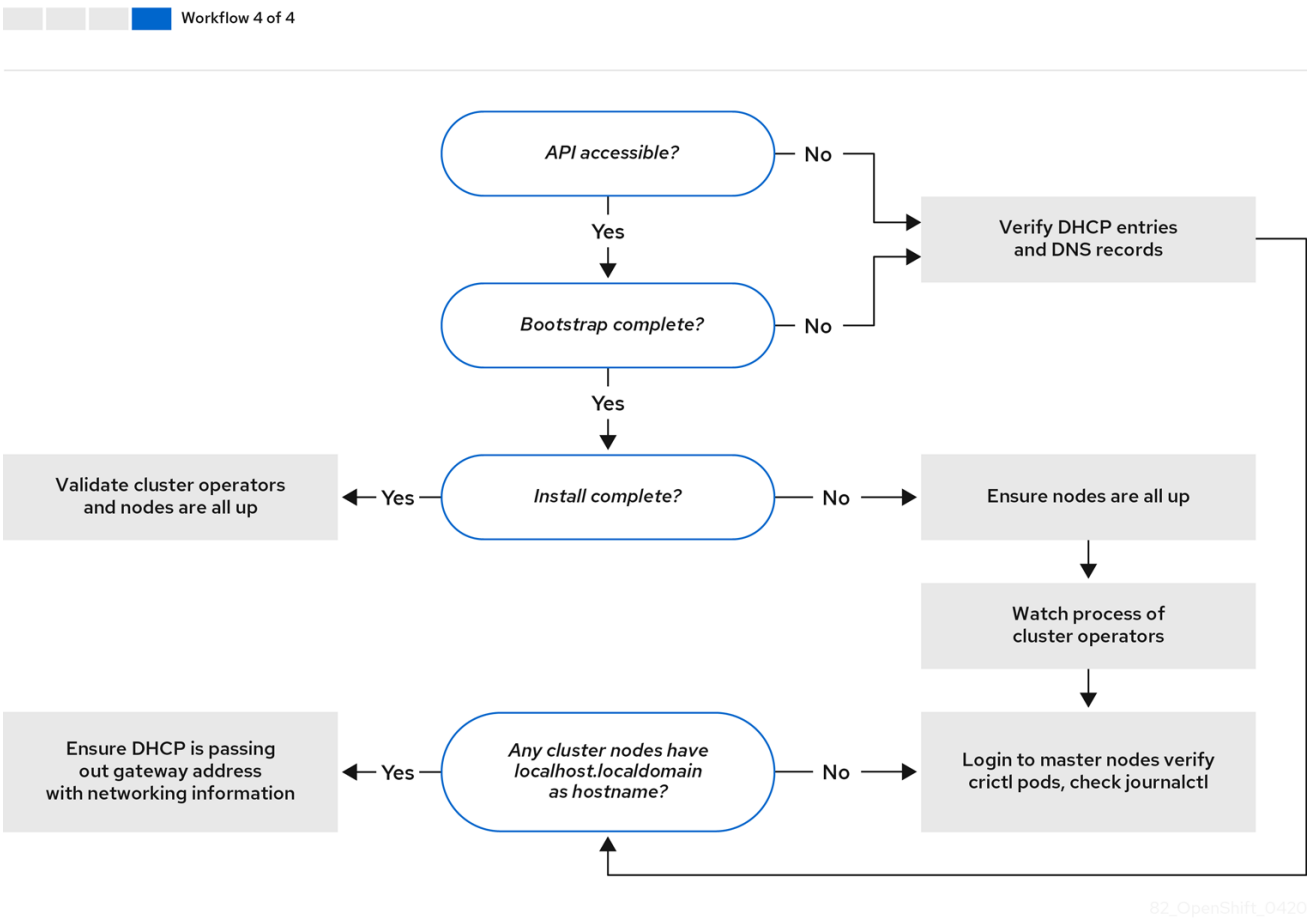
当 **install-config.yaml** 文件出错或者无法访问 Red Hat Enterprise Linux CoreOS（RHCOS）镜像时，*工作流 1（共 4 步）* 的工作流演示了故障排除工作流。故障排除建议可在 故障排除 **install-config.yaml** 中找到。



工作流 2（共 4 步） 描述了对 bootstrap 虚拟机问题、 无法引导集群节点的 bootstrap 虚拟机 以及 检查日志 的故障排除工作流。当在没有 **provisioning** 网络的情况下安装 OpenShift Container Platform 集群时，这个工作流不适用。



*工作流 3（共 4 步）*描述了 没有 PXE 引导的集群节点的故障 排除工作流。



工作流 4（共 4 步） 演示了 无法访问的 API 的故障、验证的安装 的故障排除工作流。

6.5.2. install-config.yaml 故障排除

install-config.yaml 配置文件代表作为 OpenShift Container Platform 集群一部分的所有节点。该文件包含由 **apiVersion**、**baseDomain**、**imageContentSources** 和虚拟 IP 地址组成的必要选项。如果在 OpenShift Container Platform 集群部署早期发生错误，则 **install-config.yaml** 配置文件中可能会出现错误。

流程

1. 使用 YAML-tips 中的指南。
2. 使用 syntax-check 来验证 YAML 语法是否正确。
3. 验证 Red Hat Enterprise Linux CoreOS（RHCOS）QEMU 镜像是否已正确定义，并可以通过 **install-config.yaml** 提供的 URL 访问。例如：

```
$ curl -s -o /dev/null -I -w "%{http_code}\n" http://webserver.example.com:8080/rhcos-44.81.202004250133-0-qemu.x86_64.qcow2.gz?sha256=7d884b46ee54fe87bbc3893bf2aa99af3b2d31f2e19ab5529c60636fbd0f1ce7
```

如果输出为 **200**，则代表从存储 bootstrap 虚拟机镜像的 webserver 返回了有效的响应。

6.5.3. Bootstrap 虚拟机问题

OpenShift Container Platform 安装程序生成 bootstrap 节点虚拟机，该虚拟机处理置备 OpenShift Container Platform 集群节点。

流程

1. 触发安装程序后约 10 到 15 分钟，使用 **virsh** 命令检查 bootstrap 虚拟机是否可正常工作：

```
$ sudo virsh list
```

```
 Id    Name                           State
----------------------------------------------------
 12    openshift-xf6fq-bootstrap      running
```

注意

bootstrap 虚拟机的名称始终是集群名称再加上一组随机字符，并以"bootstrap"结尾。

如果 bootstrap 虚拟机在 10 到 15 分钟后还没有运行，请检查其没有运行的原因。可能的问题包括：

2. 确定在该系统中运行了 **libvirtd**:

```
$ systemctl status libvirtd
```

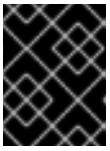
```
● libvirtd.service - Virtualization daemon
   Loaded: loaded (/usr/lib/systemd/system/libvirtd.service; enabled; vendor preset: enabled)
   Active: active (running) since Tue 2020-03-03 21:21:07 UTC; 3 weeks 5 days ago
     Docs: man:libvirtd(8)
           https://libvirt.org
 Main PID: 9850 (libvirtd)
    Tasks: 20 (limit: 32768)
   Memory: 74.8M
   CGroup: /system.slice/libvirtd.service
           ├─ 9850 /usr/sbin/libvirtd
```

如果 bootstrap 虚拟机可以正常工作，请登录它。

3. 使用 **virsh console** 命令查找 bootstrap 虚拟机的 IP 地址：

```
$ sudo virsh console example.com
```

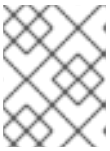
```
Connected to domain example.com
Escape character is ^]

Red Hat Enterprise Linux CoreOS 43.81.202001142154.0 (Ootpa) 4.3
SSH host key: SHA256:BRWJktXZgQQRY5zjuAV0IKZ4WM7i4TiUyMVanqu9Pqg (ED25519)
SSH host key: SHA256:7+iKGA7VtG5szmk2jB5gl/5EZ+SNcJ3a2g23o0lnIio (ECDSA)
SSH host key: SHA256:DH5VWhvhvagOTaLsYiVNse9ca+ZSW/30OOMed8rIGOc (RSA)
ens3:  fd35:919d:4042:2:c7ed:9a9f:a9ec:7
ens4: 172.22.0.2 fe80::1d05:e52e:be5d:263f
localhost login:
```



重要

当在没有 **provisioning** 网络的情况下部署 OpenShift Container Platform 集群时，您必须使用公共 IP 地址，而不是像 **172.22.0.2** 这样的私有 IP 地址。

4. 获取 IP 地址后，使用 **ssh** 命令登录到 bootstrap 虚拟机：



注意

在上一步的控制台输出中，您可以使用 **ens3** 提供的 IPv6 IP 地址或 **ens4** 提供的 IPv4 IP。

```
$ ssh core@172.22.0.2
```

如果您无法成功登录到 bootstrap 虚拟机，您可能会遇到以下情况之一：

- 无法访问 **172.22.0.0/24** 网络。验证 provisioner 主机上的网络连接，特别是 **provisioner** 网桥上的连接。如果您不使用 **provisioning** 网络，则不会有这个问题。
- 您无法通过公共网络访问 bootstrap 虚拟机。当尝试通过 **baremetal** 网络进行 SSH 时，验证 **provisioner** 主机的连接，特别是 **baremetal** 网桥的连接。
- 存在 **Permission denied (publickey,password,keyboard-interactive)** 问题。当尝试访问 bootstrap 虚拟机时，可能会出现 **Permission denied** 错误。验证试图登录到虚拟机的用户的 SSH 密钥是否在 **install-config.yaml** 文件中设置。

6.5.3.1. Bootstrap 虚拟机无法引导集群节点

在部署期间，bootstrap 虚拟机可能无法引导集群节点，这会阻止虚拟机使用 RHCOS 镜像置备节点。这可能是因为以下原因：

- **install-config.yaml** 文件有问题。
- 通过裸机网络进行带外网络访问的问题。

要验证这个问题，有三个与 **ironic** 相关的容器：

- **ironic-api**
- **ironic-conductor**
- **ironic-inspector**

流程

1. 登录到 bootstrap 虚拟机：

```
$ ssh core@172.22.0.2
```

2. 要检查容器日志，请执行以下操作：

```
[core@localhost ~]$ sudo podman logs -f <container-name>
```

将 **<container-name>** 替换为 **ironic-api**、**ironic-conductor** 或 **ironic-inspector** 之一。如果您

遇到 control plane 节点没有通过 PXE 引导的问题，请检查 **ironic-conductor** pod。**ironic-conductor** pod 包含了有关尝试引导集群节点的最详细信息，因为它尝试通过 IPMI 登录该节点。

潜在原因

集群节点在部署启动时可能处于 **ON** 状态。

解决方案

在通过 IPMI 开始安装前关闭 OpenShift Container Platform 集群节点：

```
$ ipmitool -I lanplus -U root -P <password> -H <out-of-band-ip> power off
```

6.5.3.2. 检查日志

在下载或访问 RHCOS 镜像时，首先请验证 **install-config.yaml** 配置文件中的 URL 是否正确。

内部 webserver 托管 RHCOS 镜像的示例

```
bootstrapOSImage: http://<ip:port>/rhcos-43.81.202001142154.0-qemu.x86_64.qcow2.gz?sha256=9d999f55ff1d44f7ed7c106508e5deecd04dc3c06095d34d36bf1cd127837e0c
clusterOSImage: http://<ip:port>/rhcos-43.81.202001142154.0-openstack.x86_64.qcow2.gz?sha256=a1bda656fa0892f7b936fdc6b6a6086bddaed5dafacedcd7a1e811abb78fe3b0
```

ipa-downloader 和 **coreos-downloader** 容器从 Webserver 或外部 quay.io registry 下载资源（由 **install-config.yaml** 配置文件中指定的为准）。验证以下两个容器是否正在运行，并根据需要检查其日志：

- **ipa-downloader**
- **coreos-downloader**

流程

1. 登录到 bootstrap 虚拟机：

   ```
   $ ssh core@172.22.0.2
   ```

2. 检查 bootstrap 虚拟机中的 **ipa-downloader** 和 **coreos-downloader** 容器的状态：

   ```
   [core@localhost ~]$ sudo podman logs -f ipa-downloader
   ```

   ```
   [core@localhost ~]$ sudo podman logs -f coreos-downloader
   ```

   如果 bootstrap 虚拟机无法访问镜像的 URL，使用 **curl** 命令验证虚拟机能否访问镜像。

3. 要检查 **bootkube** 日志以了解在部署阶段是否启动了所有容器，请执行以下操作：

   ```
   [core@localhost ~]$ journalctl -xe
   ```

   ```
   [core@localhost ~]$ journalctl -b -f -u bootkube.service
   ```

4. 验证所有 pod 都在运行，包括 **dnsmasq**、**mariadb**、**httpd** 和 **ironic**：

```
[core@localhost ~]$ sudo podman ps
```

5. 如果 pod 存在问题，请检查相关的容器日志。要检查 **ironic-api** 的日志，请执行以下操作：

```
[core@localhost ~]$ sudo podman logs <ironic-api>
```

6.5.4. 集群节点不能 PXE 引导

当 OpenShift Container Platform 集群节点无法 PXE 引导时，在不能 PXE 引导的集群节点上执行以下检查。如果没有 **provisioning** 网络，则在安装 OpenShift Container Platform 集群时不适用此步骤。

流程

1. 检查到 **provisioning** 网络的网络连接。
2. 确保 **provisioning** 网络的 NIC 上启用了 PXE，并且其他所有 NIC 都禁用了 PXE。
3. 验证 **install-config.yaml** 配置文件是否有正确的硬件配置集，以及连接到 **provisioning** 网络的 NIC 的引导 MAC 地址。例如：

   control plane 节点设置

```
bootMACAddress: 24:6E:96:1B:96:90 # MAC of bootable provisioning NIC
hardwareProfile: default          #control plane node settings
```

   Worker 节点设置

```
bootMACAddress: 24:6E:96:1B:96:90 # MAC of bootable provisioning NIC
hardwareProfile: unknown          #worker node settings
```

6.5.5. API 无法访问

当集群正在运行且客户端无法访问 API 时，可能是因为域名解析的问题影响对 API 的访问。

流程

1. **主机名解析：** 检查集群节点带有完全限定域名，而不只是 **localhost.localdomain**。例如：

```
$ hostname
```

   如果没有设定主机名，请设置正确的主机名。例如：

```
$ hostnamectl set-hostname <hostname>
```

2. **错误的名称解析：** 使用 **dig** 和 **nslookup** 检查每个节点在 DNS 服务器中是否可以被正确解析。例如：

```
$ dig api.<cluster-name>.example.com
```

```
; <<>> DiG 9.11.4-P2-RedHat-9.11.4-26.P2.el8 <<>> api.<cluster-name>.example.com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 37551
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 1, ADDITIONAL: 2

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
; COOKIE: 866929d2f8e8563582af23f05ec44203d313e50948d43f60 (good)
;; QUESTION SECTION:
;api.<cluster-name>.example.com. IN A

;; ANSWER SECTION:
api.<cluster-name>.example.com. 10800 IN A 10.19.13.86

;; AUTHORITY SECTION:
<cluster-name>.example.com. 10800 IN NS <cluster-name>.example.com.

;; ADDITIONAL SECTION:
<cluster-name>.example.com. 10800 IN A 10.19.14.247

;; Query time: 0 msec
;; SERVER: 10.19.14.247#53(10.19.14.247)
;; WHEN: Tue May 19 20:30:59 UTC 2020
;; MSG SIZE  rcvd: 140
```

示例中的输出显示 **api.<cluster-name>.example.com** VIP 的适当 IP 地址为 **10.19.13.86**。这个 IP 地址应该位于 **baremetal** 网络中。

6.5.6. 清理以前的安装

如果上一个部署失败，在尝试再次部署 OpenShift Container Platform 前从失败的尝试中删除工件。

流程

1. 在安装 OpenShift Container Platform 集群前关闭所有裸机节点：

```
$ ipmitool -I lanplus -U <user> -P <password> -H <management-server-ip> power off
```

2. 删除以前部署中保留的旧的 bootstrap 资源：

```
for i in $(sudo virsh list | tail -n +3 | grep bootstrap | awk {'print $2'});
do
  sudo virsh destroy $i;
  sudo virsh undefine $i;
  sudo virsh vol-delete $i --pool $i;
  sudo virsh vol-delete $i.ign --pool $i;
  sudo virsh pool-destroy $i;
  sudo virsh pool-undefine $i;
done
```

3. 从 **clusterconfigs** 目录中删除以下内容以防止 Terraform 失败：

```
$ rm -rf ~/clusterconfigs/auth ~/clusterconfigs/terraform* ~/clusterconfigs/tls
~/clusterconfigs/metadata.json
```

6.5.7. 创建 registry 的问题

在创建断开连接的 registry 时，在尝试对 registry 进行镜像时，可能会遇到 "User Not Authorized" 错误。如果您没有在现有的 **pull-secret.txt** 文件中添加新身份验证，则可能会出现这个错误。

流程

1. 检查以确保身份验证成功：

```
$ /usr/local/bin/oc adm release mirror \
  -a pull-secret-update.json
  --from=$UPSTREAM_REPO \
  --to-release-image=$LOCAL_REG/$LOCAL_REPO:${VERSION} \
  --to=$LOCAL_REG/$LOCAL_REPO
```

注意

用于镜像安装镜像的变量输出示例：

```
UPSTREAM_REPO=${RELEASE_IMAGE}
LOCAL_REG=<registry_FQDN>:<registry_port>
LOCAL_REPO='ocp4/openshift4'
```

在为 **OpenShift 安装设置环境**章节中的**获取 OpenShift 安装程序**步骤中，设置了 **RELEASE_IMAGE** 和 **VERSION** 值。

2. 镜像 registry 后，确认您可以在断开连接的环境中访问它：

```
$ curl -k -u <user>:<password> https://registry.example.com:<registry-port>/v2/_catalog
{"repositories":["<Repo-Name>"]}
```

6.5.8. 其它问题

6.5.8.1. 解决 runtime network not ready 错误

部署集群后您可能会收到以下错误：

```
`runtime network not ready: NetworkReady=false reason:NetworkPluginNotReady message:Network
plugin returns error: Missing CNI default network`
```

Cluster Network Operator 负责部署网络组件以响应安装程序创建的特殊对象。它会在安装过程的早期阶段运行（在 Control Plane（master）节点启动后，bootstrap control plane 被停止前运行）。它可能会显示更细微的安装程序问题，比如启动 Control Plane（master）节点时延迟时间过长，或者 **apiserver** 通讯的问题。

流程

1. 检查 **openshift-network-operator** 命名空间中的 pod:

```
$ oc get all -n openshift-network-operator
```

```
NAME                                READY STATUS            RESTARTS   AGE
pod/network-operator-69dfd7b577-bg89v   0/1   ContainerCreating 0          149m
```

2. 在 **provisioner** 节点上，确定存在网络配置：

```
$ kubectl get network.config.openshift.io cluster -oyaml
```

```
apiVersion: config.openshift.io/v1
kind: Network
metadata:
  name: cluster
spec:
  serviceNetwork:
  - 172.30.0.0/16
  clusterNetwork:
  - cidr: 10.128.0.0/14
    hostPrefix: 23
  networkType: OpenShiftSDN
```

如果不存在，代表安装程序没有创建它。要确定安装程序没有创建它的原因，请执行以下操作：

```
$ openshift-install create manifests
```

3. 检查 **network-operator** 是否正在运行：

```
$ kubectl -n openshift-network-operator get pods
```

4. 检索日志：

```
$ kubectl -n openshift-network-operator logs -l "name=network-operator"
```

在具有三个或更多 Control Plane(master)节点的高可用性集群上，Operator 将执行领导选举机制，所有其他 Operator 会休眠。如需了解更多详细信息，请参阅 故障排除。

6.5.8.2. 集群节点没有通过 DHCP 获得正确的 IPv6 地址

如果集群节点没有通过 DHCP 获得正确的 IPv6 地址，请检查以下内容：

1. 确定保留的 IPv6 地址不在 DHCP 范围之外。
2. 在 DHCP 服务器的 IP 地址保留中，确保保留指定了正确的 DHCP 唯一识别符（DUID）。例如：

```
# This is a dnsmasq dhcp reservation, 'id:00:03:00:01' is the client id and '18:db:f2:8c:d5:9f' is
the MAC Address for the NIC
id:00:03:00:01:18:db:f2:8c:d5:9f,openshift-master-1,[2620:52:0:1302::6]
```

3. 确保路由声明（Route Announcement）正在正常工作。
4. 确定 DHCP 服务器正在侦听提供 IP 地址范围所需的接口。

6.5.8.3. 集群节点没有通过 DHCP 获得正确的主机名

在 IPv6 部署过程中，集群节点必须通过 DHCP 获得其主机名。有时 **NetworkManager** 不会立即分配主机名。Control Plane（master）节点可能会报告错误，例如：

```
Failed Units: 2
  NetworkManager-wait-online.service
  nodeip-configuration.service
```

这个错误表示集群节点可能在没有从 DHCP 服务器收到主机名的情况下引导，这会导致 **kubelet** 使用 **localhost.localdomain** 主机名引导。要解决这个问题，强制节点更新主机名。

流程

1. 检索 **主机名**:

   ```
   [core@master-X ~]$ hostname
   ```

   如果主机名是 **localhost**，请执行以下步骤。

   注意

   其中 **X** 是 control plane 节点（也称为 master 节点）号。

2. 强制集群节点续订 DHCP 租期：

   ```
   [core@master-X ~]$ sudo nmcli con up "<bare-metal-nic>"
   ```

   将 **<bare-metal-nic>** 替换为与 **baremetal** 网络对应的有线连接。

3. 再次检查 **主机名**:

   ```
   [core@master-X ~]$ hostname
   ```

4. 如果主机名仍然是 **localhost.localdomain**，重启 **NetworkManager**:

   ```
   [core@master-X ~]$ sudo systemctl restart NetworkManager
   ```

5. 如果主机名仍然是 **localhost.localdomain**，请等待几分钟并再次检查。如果主机名还是 **localhost.localdomain**，重复前面的步骤。

6. 重启 **nodeip-configuration** 服务：

   ```
   [core@master-X ~]$ sudo systemctl restart nodeip-configuration.service
   ```

   此服务将使用正确的主机名引用来重新配置 **kubelet** 服务。

7. 因为 kubelet 在上一步中有所改变，所以重新加载单元文件定义：

   ```
   [core@master-X ~]$ sudo systemctl daemon-reload
   ```

8. 重启 **kubelet** 服务：

```
[core@master-X ~]$ sudo systemctl restart kubelet.service
```

9. 确保 **kubelet** 使用正确的主机名引导：

```
[core@master-X ~]$ sudo journalctl -fu kubelet.service
```

如果集群节点在启动并运行集群后没有通过 DHCP 获得正确的主机名（例如在重启过程中），集群将会有一个待处理的 **csr**。**不要**批准 **csr**，否则可能会出现其他问题。

处理 **csr**

1. 在集群上获取 CSR:

```
$ oc get csr
```

2. 验证待处理的 **csr** 是否包含 **Subject Name: localhost.localdomain**:

```
$ oc get csr <pending_csr> -o jsonpath='{.spec.request}' | base64 --decode | openssl req -noout -text
```

3. 删除包含 **Subject Name: localhost.localdomain** 的任何 **csr**:

```
$ oc delete csr <wrong_csr>
```

6.5.8.4. 路由无法访问端点

在安装过程中，可能会遇到虚拟路由器冗余协议（VRRP）冲突。如果以前使用一个特定集群名称部署的集群中的个 OpenShift Container Platform 节点仍在运行，且这个节点不是使用相同集群名称部署的当前 OpenShift Container Platform 集群的一部分，则可能会出现冲突。例如，一个集群使用集群名称 **openshift** 部署，它部署了三个 Control Plane（master）节点和三个 worker 节点。之后，一个单独的安装会使用相同的集群名称 **openshift**，但这个重新部署只安装了三个 control plane(master)节点，以前部署的三个 worker 节点处于 **ON** 状态。这可能导致 Virtual Router Identifier（VRID）冲突和 VRRP 冲突。

1. 获取路由：

```
$ oc get route oauth-openshift
```

2. 检查服务端点：

```
$ oc get svc oauth-openshift
```

```
NAME              TYPE        CLUSTER-IP      EXTERNAL-IP   PORT(S)   AGE
oauth-openshift   ClusterIP   172.30.19.162   <none>        443/TCP   59m
```

3. 尝试从 Control Plane（master）节点访问该服务：

```
[core@master0 ~]$ curl -k https://172.30.19.162
```

```
{
  "kind": "Status",
  "apiVersion": "v1",
```

```
"metadata": {
},
"status": "Failure",
"message": "forbidden: User \"system:anonymous\" cannot get path \"/\"",
"reason": "Forbidden",
"details": {
},
"code": 403
```

4. 找到 **provisioner** 节点的 **authentication-operator** 错误：

```
$ oc logs deployment/authentication-operator -n openshift-authentication-operator
```

```
Event(v1.ObjectReference{Kind:"Deployment", Namespace:"openshift-authentication-operator", Name:"authentication-operator", UID:"225c5bd5-b368-439b-9155-5fd3c0459d98", APIVersion:"apps/v1", ResourceVersion:"", FieldPath:""}): type: 'Normal' reason: 'OperatorStatusChanged' Status for clusteroperator/authentication changed: Degraded message changed from "IngressStateEndpointsDegraded: All 2 endpoints for oauth-server are reporting"
```

解决方案

1. 确保每个部署的集群名称都是唯一的，确保没有冲突。
2. 关闭所有不是使用相同集群名称的集群部署的一部分的节点。否则，OpenShift Container Platform 集群的身份验证 pod 可能无法成功启动。

6.5.8.5. 在 Firstboot 过程中 Ignition 失败

在 Firstboot 过程中，Ignition 配置可能会失败。

流程

1. 连接到 Ignition 配置失败的节点：

```
Failed Units: 1
  machine-config-daemon-firstboot.service
```

2. 重启 **machine-config-daemon-firstboot** 服务：

```
[core@worker-X ~]$ sudo systemctl restart machine-config-daemon-firstboot.service
```

6.5.8.6. NTP 没有同步

OpenShift Container Platform 集群的部署需要集群节点间的 NTP 时钟已同步。如果没有同步时钟，当时间差大于 2 秒时，部署可能会因为时钟偏移而失败。

流程

1. 检查集群节点的 **AGE** 的不同。例如：

```
$ oc get nodes
```

```
NAME                         STATUS   ROLES    AGE   VERSION
master-0.cloud.example.com   Ready    master   145m   v1.16.2
master-1.cloud.example.com   Ready    master   135m   v1.16.2
master-2.cloud.example.com   Ready    master   145m   v1.16.2
worker-2.cloud.example.com   Ready    worker   100m   v1.16.2
```

2. 检查因为时钟偏移导致的时间延迟。例如：

```
$ oc get bmh -n openshift-machine-api
```

```
master-1   error registering master-1  ipmi://<out-of-band-ip>
```

```
$ sudo timedatectl
```

```
               Local time: Tue 2020-03-10 18:20:02 UTC
           Universal time: Tue 2020-03-10 18:20:02 UTC
                 RTC time: Tue 2020-03-10 18:36:53
                Time zone: UTC (UTC, +0000)
System clock synchronized: no
              NTP service: active
          RTC in local TZ: no
```

处理现有集群中的时钟偏移

1. 创建 **chrony.conf** 文件并将其编码为 **base64** 字符串。例如：

```
$ cat << EOF | base 64
server <NTP-server> iburst 1
stratumweight 0
driftfile /var/lib/chrony/drift
rtcsync
makestep 10 3
bindcmdaddress 127.0.0.1
bindcmdaddress ::1
keyfile /etc/chrony.keys
commandkey 1
generatecommandkey
noclientlog
logchange 0.5
logdir /var/log/chrony
EOF
```

1 将 **<NTP-server>** 替换为 NTP 服务器的 IP 地址。复制输出。

```
[text-in-base-64]
```

2. 创建 **MachineConfig** 对象，将 **base64** 字符串替换为上一步输出中生成的 **[text-in-base-64]** 字符串。以下示例将文件添加到 Control Plane（master）节点。您可以修改 worker 节点的文件，或为 worker 角色创建额外的机器配置。

```
$ cat << EOF > ./99_masters-chrony-configuration.yaml
```

```
apiVersion: machineconfiguration.openshift.io/v1
kind: MachineConfig
metadata:
  creationTimestamp: null
  labels:
    machineconfiguration.openshift.io/role: master
  name: 99-master-etc-chrony-conf
spec:
  config:
    ignition:
      config: {}
      security:
        tls: {}
      timeouts: {}
      version: 3.1.0
    networkd: {}
    passwd: {}
    storage:
      files:
      - contents:
          source: data:text/plain;charset=utf-8;base64,[text-in-base-64] (1)
        group:
          name: root
        mode: 420
        overwrite: true
        path: /etc/chrony.conf
        user:
          name: root
  osImageURL: ""
```

(1) 将 **[text-in-base-64]** 替换为 base64 字符串。

3. 对配置文件做一个副本备份。例如：

```
$ cp 99_masters-chrony-configuration.yaml 99_masters-chrony-configuration.yaml.backup
```

4. 应用配置文件：

```
$ oc apply -f ./masters-chrony-configuration.yaml
```

5. 确定 **System clock synchronized** 的值为 **yes**：

```
$ sudo timedatectl
```

```
               Local time: Tue 2020-03-10 19:10:02 UTC
           Universal time: Tue 2020-03-10 19:10:02 UTC
                 RTC time: Tue 2020-03-10 19:36:53
                Time zone: UTC (UTC, +0000)
System clock synchronized: yes
              NTP service: active
          RTC in local TZ: no
```

要在部署前设置时钟同步，请生成清单文件并将该文件添加到 **openshift** 目录中。例如：

```
$ cp chrony-masters.yaml ~/clusterconfigs/openshift/99_masters-chrony-configuration.yaml
```

然后继续创建集群。

6.5.9. 检查安装

安装后，请确保安装程序成功部署节点和 Pod。

流程

1. 当正确安装 OpenShift Container Platform 集群节点时，在 **STATUS** 栏中会显示 **Ready**：

```
$ oc get nodes
```

```
NAME                 STATUS   ROLES          AGE  VERSION
master-0.example.com   Ready    master,worker   4h   v1.16.2
master-1.example.com   Ready    master,worker   4h   v1.16.2
master-2.example.com   Ready    master,worker   4h   v1.16.2
```

2. 确认安装程序已成功部署所有 Pod。以下命令的输出中会排除仍在运行或已完成的 pod。

```
$ oc get pods --all-namespaces | grep -iv running | grep -iv complete
```

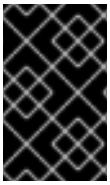
第 7 章 在 IBM Z 和 LINUXONE 上安装

7.1. 在 IBM Z 和 LINUXONE 上安装集群

在 OpenShift Container Platform 版本 4.6 中，您可以在您置备的 IBM Z 或 LinuxONE 系统上安装集群。

注意

虽然本文档只提到了 IBM Z，但它所提供的所有信息同样也适用于 LinuxONE。

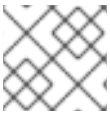


重要

非裸机平台还有其他注意事项。在尝试在此类环境中安装 OpenShift Container Platform 集群前，请参阅有关在未经测试的平台上部署 OpenShift Container Platform 的指南 中的信息。

7.1.1. 先决条件

- 在开始安装进程前，您必须清理安装目录。这可保证在安装过程中创建和更新所需的安装文件。
- 为集群置备使用 NFS 的持久性存储。若要部署私有镜像 registry，您的存储必须提供 **ReadWriteMany** 访问模式。
- 查看有关 OpenShift Container Platform 安装和更新 流程的详细信息。
- 如果使用防火墙，则必须将其配置为允许集群需要访问的站点。



注意

如果您要配置代理，请务必也要查看此站点列表。

7.1.2. OpenShift Container Platform 的互联网访问

在 OpenShift Container Platform 4.6 中，您需要访问互联网来安装集群。

您必须具有以下互联网访问权限：

- 访问 OpenShift Cluster Manager 以下载安装程序并执行订阅管理。如果集群可以访问互联网，并且没有禁用 Telemetry，该服务会自动授权您的集群。
- 访问 Quay.io，以获取安装集群所需的软件包。
- 获取执行集群更新所需的软件包。

重要

如果您的集群无法直接访问互联网，则可以在置备的某些类基础架构上执行受限网络安装。在此过程中，您要下载所需的内容，并使用它在镜像 registry（mirror registry） 中填充安装集群并生成安装程序所需的软件包。对于某些安装类型，集群要安装到的环境不需要访问互联网。在更新集群之前，要更新 registry 镜像系统中的内容。

7.1.3. 具有用户置备基础架构的集群的机器要求

对于含有用户置备的基础架构的集群，您必须部署所有所需的机器。

7.1.3.1. 所需的机器

最小的 OpenShift Container Platform 集群需要下列主机：

- 一个临时 bootstrap 机器
- 三台 control plane 或 master 机器
- 至少两台计算机器，也称为 worker 机器。

注意

集群要求 bootstrap 机器在三台 control plane 机器上部署 OpenShift Container Platform 集群。您可在安装集群后删除 bootstrap 机器。

重要

要提高集群的高可用性，请在最少两个物理集群中的不同的 z/VM 实例中分布运行 control plane 机器。

bootstrap 和 control plane 机器必须使用 Red Hat Enterprise Linux CoreOS (RHCOS) 作为操作系统。但是，计算机器可以在 Red Hat Enterprise Linux CoreOS(RHCOS)或 Red Hat Enterprise Linux(RHEL)7.9 之间进行选择。

请注意，RHCOS 基于 Red Hat Enterprise Linux（RHEL） 8，并继承其所有硬件认证和要求。请查看Red Hat Enterprise Linux 技术功能及限制 。

7.1.3.2. 网络连接要求

所有 Red Hat Enterprise Linux CoreOS (RHCOS) 机器在启动过程中需要 **initramfs** 中的网络从 Machine Config Server 获取 Ignition 配置文件。机器被配置为使用静态 IP 地址。不需要 DHCP 服务器。另外，集群中的每个 OpenShift Container Platform 节点都必须有权访问网络时间协议（NTP）服务器。

7.1.3.3. IBM Z 网络连接要求

要在 z/VM 中安装 IBM Z，您需要使用第 2 层模式的单一 z/VM 虚拟 NIC。您还需要：

- 直接连接的 OSA 或 RoCE 网络适配器
- z/VM vSwitch 设置。对于首选的设置，请使用 OSA 链接聚合。

7.1.3.4. 最低资源要求

每台集群机器都必须满足以下最低要求：

表 7.1. 最低资源要求

机器	操作系统	vCPU [1]	虚拟内存	存储	IOPS
bootstrap	RHCOS	4	16 GB	100 GB	N/A

机器	操作系统	vCPU [1]	虚拟内存	存储	IOPS
Control plane	RHCOS	4	16 GB	100 GB	N/A
Compute	RHCOS	2	8 GB	100 GB	N/A

1. 当未启用并发多线程 (SMT) 或超线程时，一个 vCPU 相当于一个物理内核。启用后，使用以下公式来计算对应的比例：（每个内核数的线程）× sockets = vCPU。

7.1.3.5. 最低 IBM Z 系统环境

您可以在以下 IBM 硬件上安装 OpenShift Container Platform 版本 4.6：

- IBM z15（所有型号）、IBM z14（所有型号）、IBM z13 和 IBM z13s
- 任何版本的 LinuxONE

硬件要求

- 为每个集群启用 SMT2 相当于 6 个 IFL。
- 至少一个网络连接连接到 **LoadBalancer** 服务，并为集群外的流量提供数据。

注意

您可以使用专用或共享的 IFL 来分配足够的计算资源。资源共享是 IBM Z 的一个关键优势。但是，您必须正确调整每个虚拟机监控程序层上的容量，并确保每个 OpenShift Container Platform 集群都有充足的资源。

重要

由于集群的整体性能可能会受到影响，用于设置 OpenShift Container Platform 集群的 LPAR 必须提供足够的计算容量。就此而言，管理程序级别上的 LPAR 权重管理、授权和 CPU 共享扮演着重要角色。

操作系统要求

- 一个 z/VM 7.1 或更高版本的实例

在您的 z/VM 实例中设置：

- 3 个客户虚拟机作为 OpenShift Container Platform control plane 的机器
- 2 个客户虚拟机作为 OpenShift Container Platform 的计算机器
- 1 个客户虚拟机作为临时 OpenShift Container Platform bootstrap 机器

IBM Z 网络连接要求
要在 z/VM 中安装 IBM Z，您需要使用第 2 层模式的单一 z/VM 虚拟 NIC。您还需要：

- 直接连接的 OSA 或 RoCE 网络适配器
- z/VM vSwitch 设置。对于首选的设置，请使用 OSA 链接聚合。

z/VM 客户虚拟机的磁盘存储

- 附加了 FICON 的磁盘存储。可以是 z/VM Minidisks、fullpack Minidisks 或专用 DASD，它们都必须被格式化为 CDL，这是默认的 CDL。要达到 Red Hat Enterprise Linux CoreOS (RHCOS) 安装所需的最小 DASD 大小，您需要扩展地址卷 (EAV)。如果可用，使用 HyperPAV 来确保最佳性能。
- FCP 连接的磁盘存储

存储/主内存

- OpenShift Container Platform control plane 需要 16 GB
- OpenShift Container Platform 计算机器需要 8 GB
- 临时 OpenShift Container Platform bootstrap 机器需要 16 GB

7.1.3.6. 首选 IBM Z 系统环境

硬件要求

- 3 个 LPARS，每个集群都有相当于 6 个 IFL 的 LPARS，它们启用了 SMT2。
- 两个网络连接连接到 **LoadBalancer** 服务，并为集群外的流量提供数据。
- HiperSockets，可以直接作为设备附加到节点，或者与一个 z/VM VSWITCH 桥接，对 z/VM 客户机透明。要将 HiperSockets 直接连接到节点，您必须通过 RHEL 8 客户机设置到外部网络的网关来桥接到 HiperSockets 网络。

操作系统要求

- 2 个或 3 个 z/VM 7.1 或更高版本的实例以实现高可用性

在您的 z/VM 实例中设置：

- 3 个用于 OpenShift Container Platform control plane 机器的虚拟机，每个 z/VM 实例一个。
- 至少 6 个用于 OpenShift Container Platform 计算机器的虚拟机，分布在 z/VM 实例中。
- 1 个客户虚拟机作为临时 OpenShift Container Platform bootstrap 机器。
- 要确保过量使用环境中组件的可用性，请使用 CP 命令 **SET SHARE** 来提高 control plane 的优先级。如果存在基础架构节点，则对它们执行相同的操作。请参阅 IBM 文档中的 SET SHARE。

IBM Z 网络连接要求
要在 z/VM 中安装 IBM Z，您需要使用第 2 层模式的单一 z/VM 虚拟 NIC。您还需要：

- 直接连接的 OSA 或 RoCE 网络适配器
- z/VM vSwitch 设置。对于首选的设置，请使用 OSA 链接聚合。

z/VM 客户虚拟机的磁盘存储

- 附加了 FICON 的磁盘存储。可以是 z/VM Minidisks、fullpack Minidisks 或专用 DASD，它们都必须被格式化为 CDL，这是默认的 CDL。要达到 Red Hat Enterprise Linux CoreOS (RHCOS) 安装所需的最小 DASD 大小，您需要扩展地址卷 (EAV)。如果可用，请使用 HyperPAV 和 High Performance FICON（zHPF）来确保最佳性能。

- FCP 连接的磁盘存储

存储/主内存

- OpenShift Container Platform control plane 需要 16 GB
- OpenShift Container Platform 计算机器需要 8 GB
- 临时 OpenShift Container Platform bootstrap 机器需要 16 GB

7.1.3.7. 证书签名请求管理

在使用您置备的基础架构时，集群只能有限地访问自动机器管理，因此您必须提供一种在安装后批准集群证书签名请求 (CSR) 的机制。**kube-controller-manager** 只能批准 kubelet 客户端 CSR。 **machine-approver** 无法保证使用 kubelet 凭证请求的提供证书的有效性，因为它不能确认是正确的机器发出了该请求。您必须决定并实施一种方法，以验证 kubelet 提供证书请求的有效性并进行批准。

其他资源

- 请参阅 IBM 文档中的使用 z/VM 虚拟交换机桥接 HiperSockets LAN。
- 请参阅在 z/VM 的 Linux 客户端中扩展 HyperPAV 别名设备 以获得性能优化。
- 有关 LPAR 权重管理和权利的信息，请参阅 LPAR 性能的主题。

7.1.4. 创建用户置备的基础架构

在部署采用用户置备的基础架构的 OpenShift Container Platform 集群前，您必须创建底层基础架构。

先决条件

- 在为集群创建支持基础架构之前，请参阅OpenShift Container Platform 4.x Tested Integrations页。

流程

1. 设置静态 IP 地址
2. 设置一个 FTP 服务器。
3. 提供所需的负载均衡器。
4. 配置机器的端口。
5. 配置 DNS。
6. 确保网络可以正常工作。

7.1.4.1. 用户置备的基础架构对网络的要求

所有 Red Hat Enterprise Linux CoreOS（RHCOS）机器在启动过程中需要 **initramfs** 中的网络从机器配置服务器获取 Ignition 配置。

在初次启动过程中，机器需要 FTP 服务器来建立网络连接，以下载其 Ignition 配置文件。

确保机器具有持久的 IP 地址和主机名。

Kubernetes API 服务器必须能够解析集群机器的节点名称。如果 API 服务器和 worker 节点位于不同的区域中，您可以配置默认 DNS 搜索区域，以便 API 服务器能够解析节点名称。另一种支持的方法是始终在节点对象和所有 DNS 请求中使用完全限定域名来指代主机。

您必须配置机器间的网络连接，以便集群组件进行通信。每台机器都必须能够解析集群中所有其他机器的主机名。

表 7.2. 所有机器到所有机器

协议	端口	描述
ICMP	N/A	网络可访问性测试
TCP	**1936**	指标
	9000-**9999**	主机级别的服务，包括端口 **9100**-**9101** 上的节点导出器和端口 **9099** 上的 Cluster Version Operator。
	10250-**10259**	Kubernetes 保留的默认端口
	10256	openshift-sdn
UDP	**4789**	VXLAN 和 Geneve
	6081	VXLAN 和 Geneve
	9000-**9999**	主机级别的服务，包括端口 **9100**-**9101** 上的节点导出器。
TCP/UDP	**30000**-**32767**	Kubernetes 节点端口

表 7.3. 要通过控制平面的所有机器

协议	端口	描述
TCP	**6443**	Kubernetes API

表 7.4. control plane 机器到 control plane 机器

协议	端口	描述
TCP	**2379**-**2380**	etcd 服务器和对等端口

网络拓扑要求

您为集群置备的基础架构必须满足下列网络拓扑要求。

重要

OpenShift Container Platform 要求所有节点都能访问互联网，以便为平台容器提取镜像并向红帽提供遥测数据。

负载均衡器

在安装 OpenShift Container Platform 前，您必须置备两个满足以下要求的负载均衡器：

1. **API 负载均衡器**：提供一个通用端点，供用户（包括人和机器）与平台交互和配置。配置以下条件：

   - 只适用于第 4 层负载均衡。这可被称为 Raw TCP、SSL Passthrough 或者 SSL 桥接模式。如果使用 SSL Bridge 模式，必须为 API 路由启用 Server Name Indication（SNI）。
   - 无状态负载平衡算法。这些选项根据负载均衡器的实现而有所不同。

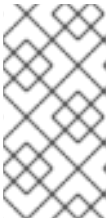
重要

不要为 API 负载均衡器配置会话持久性。

在负载均衡器的前端和后台配置以下端口：

表 7.5. API 负载均衡器

端口	后端机器（池成员）	内部	外部	描述
6443	Bootstrap 和 control plane.bootstrap 机器初始化集群 control plane 后，您要从负载均衡器中删除 bootstrap 机器。您必须为 API 服务器健康检查探测配置 **/readyz** 端点。	X	X	Kubernetes API 服务器
22623	Bootstrap 和 control plane.bootstrap 机器初始化集群 control plane 后，您要从负载均衡器中删除 bootstrap 机器。	X		机器配置服务器



注意

负载均衡器必须配置为，从 API 服务器关闭 **/readyz** 端点到从池中删除 API 服务器实例时最多需要 30 秒。在 **/readyz** 返回错误或处于健康状态后的时间范围内，端点必须被删除或添加。每 5 秒或 10 秒探测一次，有两个成功请求处于健康状态，三个成为不健康的请求经过测试。

2. **应用程序入口负载均衡器**:提供来自集群外部的应用程序流量流量的 Ingress 点。配置以下条件：

   - 只适用于第 4 层负载均衡。这可被称为 Raw TCP、SSL Passthrough 或者 SSL 桥接模式。如果使用 SSL Bridge 模式，您必须为 Ingress 路由启用Server Name Indication（SNI）。
   - 建议根据可用选项以及平台上托管的应用程序类型，使用基于连接的或者基于会话的持久性。

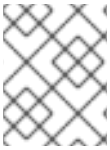
在负载均衡器的前端和后台配置以下端口：

表 7.6. 应用程序入口负载均衡器

端口	后端机器（池成员）	内部	外部	描述
443	默认运行入口路由器 Pod、计算或 worker 的机器。	X	X	HTTPS 流量
80	默认运行入口路由器 Pod、计算或 worker 的机器。	X	X	HTTP 流量

提示

如果负载均衡器可以看到客户端的真实 IP 地址，启用基于 IP 的会话持久性可提高使用端到端 TLS 加密的应用程序的性能。



注意

OpenShift Container Platform 集群需要正确配置入口路由器。control plane 初始化后，您必须配置入口路由器。

NTP 配置

OpenShift Container Platform 集群默认配置为使用公共网络时间协议（NTP）服务器。如果要使用本地企业 NTP 服务器，或者集群部署在断开连接的网络中，您可以将集群配置为使用特定的时间服务器。如需更多信息，请参阅*配置 chrony 时间服务*的文档。

其他资源

- 配置 chrony 时间服务

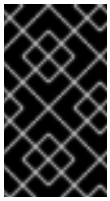
7.1.4.2. 用户置备 DNS 要求

DNS 用于名称解析和反向名称解析。DNS A/AAAA 或 CNAME 记录用于名称解析，PTR 记录用于反向解析名称。反向记录很重要，因为 Red Hat Enterprise Linux CoreOS（RHCOS）使用反向记录为所有节点设置主机名。另外，反向记录用于生成 OpenShift Container Platform 需要操作的证书签名请求（CSR）。

采用用户置备的基础架构的 OpenShift Container Platform 集群需要以下 DNS 记录。在每一记录中，**<cluster_name>** 是集群名称，**<base_domain>** 则是您在 **install-config.yaml** 文件中指定的集群基域。完整的 DNS 记录采用如下格式: **<component>.<cluster_name>.<base_domain>.**。

表 7.7. 所需的 DNS 记录

组件	记录	描述
Kubernetes API	**api.<cluster_name>.<base_domain>.**	添加 DNS A/AAAA 或 CNAME 记录，以及 DNS PTR 记录，以识别 control plane 机器的负载均衡器。这些记录必须由集群外的客户端以及集群中的所有节点解析。

组件	记录	描述
	api-int.<cluster_name>. <base_domain>.	添加 DNS A/AAAA 或 CNAME 记录，以及 DNS PTR 记录，以识别 control plane 机器的负载均衡器。这些记录必须可以从集群中的所有节点解析。  **重要** API 服务器必须能够根据在 Kubernetes 中记录的主机名解析 worker 节点。如果 API 服务器无法解析节点名称，则代理的 API 调用会失败，且您无法从 pod 检索日志。
Routes	***.apps.<cluster_name>. <base_domain>.**	添加通配符 DNS A/AAAA 或 CNAME 记录，指向以运行入口路由器 Pod 的机器（默认为 worker 节点）为目标的负载均衡器。这些记录必须由集群外的客户端以及集群中的所有节点解析。
bootstrap	**bootstrap.<cluster_name>. <base_domain>.**	添加 DNS A/AAAA 或 CNAME 记录，以及 DNS PTR 记录来识别 bootstrap 机器。这些记录必须由集群中的节点解析。
Master 主机	**<master><n>. <cluster_name>. <base_domain>.**	DNS A/AAAA 或 CNAME 记录，以识别 control plane 节点（也称为 master 节点）的每台机器。这些记录必须由集群中的节点解析。
Worker 主机	**<worker><n>. <cluster_name>. <base_domain>.**	添加 DNS A/AAAA 或 CNAME 记录，以识别 worker 节点的每台机器。这些记录必须由集群中的节点解析。

提示

您可以使用 **nslookup <hostname>** 命令来验证名称解析。您可以使用 **dig -x <ip_address>** 命令来验证 PTR 记录的反向名称解析。

下面的 BIND 区文件的例子展示了关于名字解析的 A 记录的例子。这个示例的目的是显示所需的记录。这个示例不是为选择一个名称解析服务提供建议。

例 7.1. DNS 区数据库示例

```
$TTL 1W
@ IN SOA ns1.example.com. root (
   2019070700 ; serial
   3H  ; refresh (3 hours)
   30M  ; retry (30 minutes)
   2W  ; expiry (2 weeks)
   1W )  ; minimum (1 week)
 IN NS ns1.example.com.
 IN MX 10 smtp.example.com.
;
;
```

```
ns1 IN A 192.168.1.5
smtp IN A 192.168.1.5
;
helper IN A 192.168.1.5
helper.ocp4 IN A 192.168.1.5
;
; The api identifies the IP of your load balancer.
api.ocp4  IN A 192.168.1.5
api-int.ocp4  IN A 192.168.1.5
;
; The wildcard also identifies the load balancer.
*.apps.ocp4  IN A 192.168.1.5
;
; Create an entry for the bootstrap host.
bootstrap.ocp4 IN A 192.168.1.96
;
; Create entries for the master hosts.
master0.ocp4  IN A 192.168.1.97
master1.ocp4  IN A 192.168.1.98
master2.ocp4  IN A 192.168.1.99
;
; Create entries for the worker hosts.
worker0.ocp4  IN A 192.168.1.11
worker1.ocp4  IN A 192.168.1.7
;
;EOF
```

下面的 BIND 区文件示例显示了反向名字解析的 PTR 记录示例。

例 7.2. 反向记录的 DNS 区数据库示例

```
$TTL 1W
@ IN SOA ns1.example.com. root (
   2019070700 ; serial
   3H  ; refresh (3 hours)
   30M  ; retry (30 minutes)
   2W  ; expiry (2 weeks)
   1W )  ; minimum (1 week)
 IN NS ns1.example.com.
;
; The syntax is "last octet" and the host must have an FQDN
; with a trailing dot.
97 IN PTR master0.ocp4.example.com.
98 IN PTR master1.ocp4.example.com.
99 IN PTR master2.ocp4.example.com.
;
96 IN PTR bootstrap.ocp4.example.com.
;
5 IN PTR api.ocp4.example.com.
5 IN PTR api-int.ocp4.example.com.
;
11 IN PTR worker0.ocp4.example.com.
```

```
7 IN PTR worker1.ocp4.example.com.
;
;EOF
```

7.1.5. 生成 SSH 私钥并将其添加到代理中

如果要在集群上执行安装调试或灾难恢复，则必须为 **ssh-agent** 和安装程序提供 SSH 密钥。您可以使用此密钥访问公共集群中的 bootstrap 机器来排除安装问题。

注意

在生产环境中，您需要进行灾难恢复和调试。

重要

不要在生产环境中跳过这个过程，因为生产环境需要灾难恢复和调试。

您可以使用此密钥以 **core** 用户身份通过 SSH 连接到 master 节点。在部署集群时，此密钥会添加到 **core** 用户的 **~/.ssh/authorized_keys** 列表中。

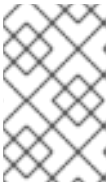
流程

1. 如果还没有为计算机上免密码身份验证而配置的 SSH 密钥，请创建一个。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ ssh-keygen -t ed25519 -N '' \
  -f <path>/<file_name> 1
```

1 指定新 SSH 密钥的路径和文件名，如 **~/.ssh/id_rsa**。如果您已有密钥对，请确保您的公钥位于 **~/.ssh** 目录中。

运行此命令会在指定的位置生成不需要密码的 SSH 密钥。



注意

如果您计划在 **x86_64** 架构中安装使用 FIPS 验证的/Modules in Process 加密库的 OpenShift Container Platform 集群，不要创建使用 **ed25519** 算法的密钥。反之，创建一个使用 **rsa** 或 **ecdsa** 算法的密钥。

2. 作为后台任务启动 **ssh-agent** 进程：

```
$ eval "$(ssh-agent -s)"
```

输出示例

```
Agent pid 31874
```

注意

如果您的集群采用 FIPS 模式，则只使用 FIPS 兼容算法来生成 SSH 密钥。密钥必须是 RSA 或 ECDSA。

3. 将 SSH 私钥添加到 **ssh-agent**：

```
$ ssh-add <path>/<file_name> 1
```

输出示例

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

1 指定 SSH 私钥的路径和文件名，如 **~/.ssh/id_rsa**

后续步骤

- 在安装 OpenShift Container Platform 时，为安装程序提供 SSH 公钥。

7.1.6. 获取安装程序

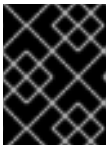
在安装 OpenShift Container Platform 之前，将安装文件下载到您置备的机器上。

先决条件

- 一个运行 Linux 的机器，如 Red Hat Enterprise Linux 8，本地磁盘空间为 500MB。

流程

1. 访问 OpenShift Cluster Manager 站点的 Infrastructure Provider 页面。如果您有红帽帐号，请使用自己的凭证登录。如果没有，请创建一个帐户。
2. 选择您的基础架构供应商。
3. 进入适用于您的安装类型的页面，下载您的操作系统的安装程序，并将文件放在要保存安装配置文件的目录中。。



重要

安装程序会在用来安装集群的计算机上创建若干文件。在完成集群安装后，您必须保留安装程序和安装程序所创建的文件。这两个文件都需要删除集群。

重要

删除安装程序创建的文件不会删除您的集群，即使集群在安装过程中失败也是如此。要删除集群，为特定云供应商完成 OpenShift Container Platform 卸载流程。

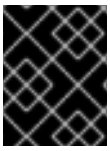
4. 提取安装程序。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ tar xvf openshift-install-linux.tar.gz
```

5. 从 Red Hat OpenShift Cluster Manager 下载安装 pull secret 。通过此 pull secret，您可以进行所含授权机构提供的服务的身份验证，这些服务包括为 OpenShift Container Platform 组件提供容器镜像的 Quay.io。

7.1.7. 通过下载二进制文件安装 OpenShift CLI

您需要安装 CLI（**oc**） 来使用命令行界面与 OpenShift Container Platform 进行交互。您可在 Linux 、Windows 或 macOS 上安装 **oc**。



重要

如果安装了旧版本的 **oc**，则无法使用 OpenShift Container Platform 4.6 中的所有命令。下载并安装新版本的 **oc**。

7.1.7.1. 在 Linux 上安装 OpenShift CLI

您可以按照以下流程在 Linux 上安装 OpenShift CLI（**oc**）二进制文件。

流程

1. 进入到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 在 **Version** 下拉菜单中选择相应的版本。
3. 单击 **OpenShift v4.6 Linux 客户端** 条目旁边的 **Download Now**，再保存文件。
4. 解包存档：

```
$ tar xvzf <file>
```

5. 把 **oc** 二进制代码放到 **PATH** 中的目录中。
执行以下命令可以查看当前的 **PATH** 设置：

```
$ echo $PATH
```

安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
$ oc <command>
```

7.1.7.2. 在 Windows 上安装 OpenShift CLI

您可以按照以下流程在 Windows 上安装 OpenShift CLI（**oc**）二进制代码。

流程

1. 进入到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 在 **Version** 下拉菜单中选择相应的版本。
3. 单击 **OpenShift v4.6 Windows 客户端**条目旁边的 **Download Now**，再保存文件。
4. 使用 ZIP 程序解压存档。

5. 把 **oc** 二进制代码放到 **PATH** 中的目录中。
要查看您的 **PATH**，请打开命令提示窗口并执行以下命令：

```
C:\> path
```

安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
C:\> oc <command>
```

7.1.7.3. 在 macOS 上安装 OpenShift CLI

您可以按照以下流程在 macOS 上安装 OpenShift CLI（**oc**）二进制代码。

流程

1. 进入到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 在 **Version** 下拉菜单中选择相应的版本。
3. 单击 **OpenShift v4.6 MacOSX 客户端**条目旁边的 **Download Now**，再保存文件。
4. 解包和解压存档。
5. 将 **oc** 二进制文件移到 PATH 的目录中。
要查看您的 **PATH**，打开一个终端窗口并执行以下命令：

```
$ echo $PATH
```

安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
$ oc <command>
```

7.1.8. 手动创建安装配置文件

对于使用用户置备的基础架构的 OpenShift Container Platform 安装，您必须手动生成安装配置文件。

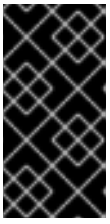
先决条件

- 获取 OpenShift Container Platform 安装程序和集群的访问令牌。

流程

1. 创建用来存储您所需的安装资产的安装目录：

```
$ mkdir <installation_directory>
```



重要

您必须创建目录。一些安装信息，如 bootstrap X.509 证书，有较短的过期间隔，因此不要重复使用安装目录。如果要重复使用另一个集群安装中的个别文件，可以将其复制到您的目录中。但是，一些安装数据的文件名可能会在发行版本之间有所改变。从 OpenShift Container Platform 老版本中复制安装文件时要格外小心。

2. 自定义以下 **install-config.yaml** 文件模板，并将它保存到 **<installation_directory>** 中。

注意

此配置文件必须命名为 **install-config.yaml**。

3. 备份 **install-config.yaml** 文件，以便用于安装多个集群。

重要

install-config.yaml 文件会在安装过程的下一步骤中消耗掉。现在必须备份它。

7.1.8.1. 安装配置参数

在部署 OpenShift Container Platform 集群前，您可以提供参数值，以描述托管集群的云平台的帐户并选择性地自定义集群平台。在创建 **install-config.yaml** 安装配置文件时，您可以通过命令行来提供所需的参数的值。如果要自定义集群，可以修改 **install-config.yaml** 文件来提供关于平台的更多信息。

注意

安装之后，您无法修改 **install-config.yaml** 文件中的这些参数。

重要

openshift-install 命令不验证参数的字段名称。如果指定了不正确的名称，则不会创建相关的文件或对象，且不会报告错误。确保所有指定的参数的字段名称都正确。

7.1.8.1.1. 所需的配置参数

下表描述了所需的安装配置参数：

表 7.8. 所需的参数

参数	描述	值
apiVersion	**install-config.yaml** 内容的 API 版本。当前版本是 **v1**。安装程序还可能支持旧的 API 版本。	字符串
baseDomain	云供应商的基域。此基础域用于创建到 OpenShift Container Platform 集群组件的路由。集群的完整 DNS 名称是 **baseDomain** 和 **metadata.name** 参数值的组合，其格式为 **<metadata.name>.<baseDomain>**。	完全限定域名或子域名，如 **example.com**。

参数	描述	值
metadata	Kubernetes 资源 **ObjectMeta**，其中只消耗 **name** 参数。	对象
metadata.name	集群的名称。集群的 DNS 记录是 **{{.metadata.name}}.{{.baseDomain}}** 的子域。	小写字母,连字符(-)和句点(.)的字符串，如 **dev**。
platform	执行安装的具体平台配置：**aws**、**baremetal**、**azure**、**openstack**、**ovirt**、**vsphere**。有关 **platform.<platform>** 参数的额外信息，请参考下表来了解您的具体平台。	对象

pullSecret	从 Red Hat OpenShift Cluster Manager 获取 pull secret，验证从 Quay.io 等服务中下载 OpenShift Container Platform 组件的容器镜像。	<pre>{
		  "auths":{
		  "cloud.openshift.com":{
		  "auth":"b3Blb=",
		  "email":"you@example.com"
		  },
		  "quay.io":{
		  "auth":"b3Blb=",
		  "email":"you@example.com"
		  }
		  }
		}</pre>

7.1.8.1.2. 网络配置参数

您可以根据现有网络基础架构的要求自定义安装配置。例如，您可以扩展集群网络的 IP 地址块，或者提供不同于默认值的不同 IP 地址块。

只支持 IPv4 地址。

表 7.9. 网络参数

参数	描述	值
networking	集群网络的配置。	对象 **注意** 您不能在安装后修改 **networking** 对象指定的参数。

参数	描述	值
networking.networkType	要安装的集群网络供应商 Container Network Interface（CNI）插件。	**OpenShiftSDN** 或 **OVNKubernetes**。默认值为 **OpenShiftSDN**。
networking.clusterNetwork	pod 的 IP 地址块。 默认值为 **10.128.0.0/14**，主机前缀为 **/23**。 如果您指定多个 IP 地址块，则块不得互相重叠。	一个对象数组。例如： networking: clusterNetwork: - cidr: 10.128.0.0/14 hostPrefix: 23
networking.clusterNetwork.cidr	使用 **networking.clusterNetwork** 时需要此项。IP 地址块。 一个 IPv4 网络。	使用 CIDR 形式的 IP 地址块。IPv4 块的前缀长度介于 **0** 到 **32** 之间。
networking.clusterNetwork.hostPrefix	分配给每个单独节点的子网前缀长度。例如，如果 **hostPrefix** 设为 **23**，则每个节点从所给的 **cidr** 中分配一个 **/23** 子网。**hostPrefix** 值 **23** 提供 510（2^(32 - 23)- 2）个 pod IP 地址。	子网前缀。 默认值为 **23**。
networking.serviceNetwork	服务的 IP 地址块。默认值为 **172.30.0.0/16**。 OpenShift SDN 和 OVN-Kubernetes 网络供应商只支持服务网络的一个 IP 地址块。	CIDR 格式具有 IP 地址块的数组。例如： networking: serviceNetwork: - 172.30.0.0/16
networking.machineNetwork	机器的 IP 地址块。 如果您指定多个 IP 地址块，则块不得互相重叠。 如果您指定了多个 IP 内核参数，**machineNetwork.cidr** 值必须是主网络的 CIDR。	一个对象数组。例如： networking: machineNetwork: - cidr: 10.0.0.0/16

参数	描述	值
networking.machineNetwork.cidr	使用 **networking.machineNetwork** 时需要。IP 地址块。libvirt 以外的所有平台的默认值为 **10.0.0.0/16**。对于 libvirt，默认值为 **192.168.126.0/24**。	CIDR 表示法中的 IP 网络块。 例如： **10.0.0.0/16**。  **注意** 将 **networking.machineNetwork** 设置为与首选 NIC 所在的 CIDR 匹配。

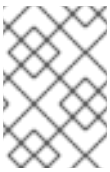
7.1.8.1.3. 可选配置参数

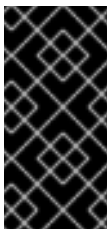
下表描述了可选安装配置参数：

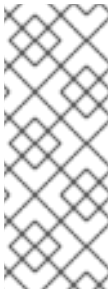
表 7.10. 可选参数

参数	描述	值
additionalTrustBundle	添加到节点可信证书存储中的 PEM 编码 X.509 证书捆绑包。配置了代理时，也可以使用这个信任捆绑包。	字符串
compute	组成计算节点的机器的配置。	machine-pool 对象的数组。详情请查看以下"Machine-pool"表。
compute.architecture	决定池中机器的指令集合架构。目前不支持异构集群，因此所有池都必须指定相同的架构。有效值为 **amd64** （默认值）。	字符串
compute.hyperthreading	是否在计算机器上启用或禁用并发多线程或**超线程**。默认情况下，启用并发多线程以提高机器内核的性能。  **重要** 如果禁用并发多线程，请确保在容量规划时考虑到机器性能可能会显著降低的问题。	**Enabled** 或 **Disabled**
compute.name	使用 **compute** 时需要此值。机器池的名称。	**worker**

参数	描述	值
compute.platform	使用 **compute** 时需要此值。使用此参数指定托管 worker 机器的云供应商。此参数值必须与 **controlPlane.platform** 参数值匹配。	**aws**、**azure**、**gcp**、**openstack**、**ovirt**、**vsphere** 或 **{}**
compute.replicas	要置备的计算机器数量，也称为 worker 机器。	大于或等于 **2** 的正整数。默认值为 **3**。
controlPlane	组成 control plane 的机器的配置。	**MachinePool** 对象的数组。详情请查看以下"Machine-pool"表。
controlPlane.architecture	决定池中机器的指令集合架构。目前不支持异构集群，因此所有池都必须指定相同的架构。有效值为 **amd64** （默认值）。	字符串
controlPlane.hyperthreading	是否在 control plane 机器上启用或禁用并发多线程或**超线程**。默认情况下，启用并发多线程以提高机器内核的性能。  **重要** 如果禁用并发多线程，请确保在容量规划时考虑到机器性能可能会显著降低的问题。	**Enabled** 或 **Disabled**
controlPlane.name	使用 **controlPlane** 时需要。机器池的名称。	**master**
controlPlane.platform	使用 **controlPlane** 时需要。使用此参数指定托管 control plane 机器的云供应商。此参数值必须与 **compute.platform** 参数值匹配。	**aws**、**azure**、**gcp**、**openstack**、**ovirt**、**vsphere** 或 **{}**
controlPlane.replicas	要置备的 control plane 机器数量。	唯一支持的值是 **3**，它是默认值。

参数	描述	值
credentialsMode	Cloud Credential Operator（CCO）模式。如果没有指定任何模式，CCO 会动态地尝试决定提供的凭证的功能，在支持多个模式的平台上使用 mint 模式。  **注意** 不是所有 CCO 模式都支持所有云供应商。如需有关 CCO 模式的更多信息，请参阅*Red Hat Operator 参考指南*内容中的 *Cloud Credential Operator* 条目。	**Mint**、**Passthrough**、**Manual** 或空字符串(**""**)。
fips	启用或禁用 FIPS 模式。默认为 **false**（禁用）。如果启用了 FIPS 模式，运行 OpenShift Container Platform 的 Red Hat Enterprise Linux CoreOS (RHCOS) 机器会绕过默认的 Kubernetes 加密套件，并使用由 RHCOS 提供的加密模块。  **重要** 只有在 **x86_64** 架构中的 OpenShift Container Platform 部署支持 FIPS 验证的/Modules in Process 加密库。  **注意** 如果使用 Azure File 存储，则无法启用 FIPS 模式。	**false** 或 **true**
imageContentSources	release-image 内容的源和仓库。	对象数组。包括一个 **source** 以及可选的 **mirrors**，如下表所示。
imageContentSources.source	使用 **imageContentSources** 时需要。指定用户在镜像拉取规格中引用的仓库。	字符串
imageContentSources.mirrors	指定可能还包含同一镜像的一个或多个仓库。	字符串数组

参数	描述	值
publish	如何发布或公开集群的面向用户的端点，如 Kubernetes API、OpenShift 路由。	**Internal** 或 **External**。默认值为 **External**。 在非云平台上不支持将此字段设置为 **Internal**。  **重要** 如果将字段的值设为 **Internal**，集群将无法运行。如需更多信息，请参阅 BZ#1953035。

sshKey	用于验证集群机器访问的 SSH 密钥或密钥。
	
	 
	注意
	
	对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 **ssh-agent** 进程使用的 SSH 密钥。
		一个或多个密钥。例如：
		
		`sshKey:`
		`  <key1>`
		`  <key2>`
		`  <key3>`

7.1.8.2. IBM Z 的 install-config.yaml 文件示例

您可以自定义 **install-config.yaml** 文件，以指定有关 OpenShift Container Platform 集群平台的更多信息，或修改所需参数的值。

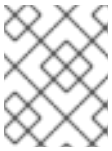
```
apiVersion: v1
baseDomain: example.com (1)
compute: (2)
- hyperthreading: Enabled (3)
  name: worker
  replicas: 0 (4)
  architecture : s390x
controlPlane: (5)
  hyperthreading: Enabled (6)
  name: master
  replicas: 3 (7)
  architecture : s390x
metadata:
```

```
  name: test 8
networking:
  clusterNetwork:
  - cidr: 10.128.0.0/14 9
    hostPrefix: 23 10
  networkType: OpenShiftSDN
  serviceNetwork: 11
  - 172.30.0.0/16
platform:
  none: {} 12
fips: false 13
pullSecret: '{"auths": ...}' 14
sshKey: 'ssh-ed25519 AAAA...' 15
```

1 集群的基域。所有 DNS 记录都必须是这个基域的子域，并包含集群名称。

2 5 **controlPlane** 部分是一个单映射，但 **compute** 部分是一系列映射。为满足不同数据结构的要求，**compute** 部分的第一行必须以连字符 **-** 开头，**controlPlane** 部分的第一行则不可以连字符开头。只使用一个 control plane 池。

3 6 是否要启用或禁用并发多线程（SMT）或**超线程**。默认情况下，启用 SMT 可提高机器内核的性能。您可以通过将参数值设为 **Disabled** 来禁用。如果禁用 SMT，则必须在所有集群机器中禁用它，其中包括 control plane 和计算机器。



注意

默认启用并发多线程（SMT）。如果在 BIOS 设置中没有启用 SMT，**hyperthreading** 参数不会起作用。

重要

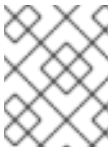
如果您禁用 **hyperthreading**（无论是在 BIOS 中还是在 **install-config.yaml** 中），请确保您对可能会造成的机器性能显著降低的情况有所考虑。

4 **replicas** 参数的值必须设置为 **0**。此参数控制集群为您创建和管理的 worker 数量，使用用户置备的基础架构时集群不会执行这些功能。在完成 OpenShift Container Platform 安装前，您必须手动为集群部署 worker 机器。

7 您添加到集群的 control plane 机器数量。由于集群将这个值用作集群中 etcd 端点的数量，因此该值必须与您部署的 control plane 机器数量匹配。

8 您在 DNS 记录中指定的集群名称。

9 从中分配 pod IP 地址的 IP 地址块。此块不得与现有的物理网络重叠。这些 IP 地址用于 pod 网络。如果您需要从外部网络访问 pod，请配置负载均衡器和路由器来管理流量。



注意

类 E CIDR 范围保留给以后使用。要使用 Class E CIDR 范围，您必须确保您的网络环境接受 Class E CIDR 范围内的 IP 地址。

10 分配给每个单独节点的子网前缀长度。例如，如果 **hostPrefix** 设为 **23**，则每个节点从所给的 **cidr**

11 用于服务 IP 地址的 IP 地址池。您只能输入一个 IP 地址池。此块不得与现有的物理网络重叠。如果您需要从外部网络访问服务，请配置负载均衡器和路由器来管理流量。

12 您必须将平台设置为 **none**。您无法为 IBM Z 基础架构提供额外的平台配置变量。

13 是否启用或禁用 FIPS 模式。默认情况下不启用 FIPS 模式。如果启用了 FIPS 模式，运行 OpenShift Container Platform 的 Red Hat Enterprise Linux CoreOS (RHCOS) 机器会绕过默认的 Kubernetes 加密套件，并使用由 RHCOS 提供的加密模块。

重要

只有在 **x86_64** 架构中的 OpenShift Container Platform 部署支持 FIPS 验证的/Modules in Process 加密库。

14 Red Hat OpenShift Cluster Manager 中的 pull secret 。通过此 pull secret，您可以进行所含授权机构提供的服务的身份验证，这些服务包括为 OpenShift Container Platform 组件提供容器镜像的 Quay.io。

15 Red Hat Enterprise Linux CoreOS (RHCOS) 中 **core** 用户的默认 SSH 密钥的公钥部分。

注意

对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 **ssh-agent** 进程使用的 SSH 密钥。

7.1.9. 在安装过程中配置集群范围代理

生产环境可能会拒绝直接访问互联网，而是提供 HTTP 或 HTTPS 代理。您可以通过在 **install-config.yaml** 文件中配置代理设置，将新的 OpenShift Container Platform 集群配置为使用代理。

先决条件

- 您有一个现有的 **install-config.yaml** 文件。
- 您检查了集群需要访问的站点，并决定是否需要绕过代理。默认情况下代理所有集群出口流量，包括对托管云供应商 API 的调用。您需要将站点添加到 **Proxy** 对象的 **spec.noProxy** 字段来绕过代理。

注意

Proxy 对象 **status.noProxy** 字段使用安装配置中的 **networking.machineNetwork[].cidr**、**networking.clusterNetwork[].cidr** 和 **networking.serviceNetwork[]** 字段的值填充。

对于在 Amazon Web Services(AWS)、Google Cloud Platform(GCP)、Microsoft Azure 和 Red Hat OpenStack Platform(RHOSP)上安装，**Proxy** 对象 **status.noProxy** 字段也会使用实例元数据端点填充(**169.254.169.254**)。

流程

1. 编辑 **install-config.yaml** 文件并添加代理设置。例如：

```
apiVersion: v1
```

```
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: example.com 3
additionalTrustBundle: | 4
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
...
```

1. 用于创建集群外 HTTP 连接的代理 URL。URL 必须是 **http**。
2. 用于创建集群外 HTTPS 连接的代理 URL。
3. 要排除在代理中的目标域名、IP 地址或其他网络 CIDR 的逗号分隔列表。在域前面加 **.** 来仅匹配子域。例如： **.y.com** 匹配 **x.y.com**，但不匹配 **y.com**。使用 * 绕过所有目的地的代理。
4. 如果提供，安装程序会在 **openshift-config 命名空间中生成名为 user-ca- bundle** 的配置映射来保存额外的 CA 证书。如果您提供 **additionalTrustBundle** 和至少一个代理设置，则 **Proxy** 对象会被配置为引用 **trustedCA** 字段中的 **user-ca-bundle** 配置映射。然后，Cluster Network Operator 会创建一个 **trusted-ca-bundle** 配置映射，该配置映射将为 **trustedCA** 参数指定的内容与 RHCOS 信任捆绑包合并。 **additionalTrustBundle** 字段是必需的，除非代理的身份证书由来自 RHCOS 信任捆绑包的颁发机构签名。

注意

安装程序不支持代理的 **readinessEndpoints** 字段。

2. 保存该文件，并在安装 OpenShift Container Platform 时引用。

安装程序会创建一个名为 **cluster** 的集群范围代理，该代理使用提供的 **install-config.yaml** 文件中的代理设置。如果没有提供代理设置，仍然会创建一个 **cluster Proxy** 对象，但它会有一个空 **spec**。

注意

只支持名为 **cluster** 的 **Proxy** 对象，且无法创建额外的代理。

7.1.10. 创建 Kubernetes 清单和 Ignition 配置文件

由于您必须修改一些集群定义文件并要手动启动集群机器，因此您必须生成 Kubernetes 清单和 Ignition 配置文件，集群需要这两项来创建其机器。

安装配置文件转换为 Kubernetes 清单。清单嵌套到 Ignition 配置文件中，稍后用于创建集群。

重要

- 安装程序生成的 Ignition 配置文件包含在 24 小时后过期的证书，然后在过期时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外情况是，您需要手动批准待处理的 **node-bootstrapper** 证书签名请求（CSR）来恢复 kubelet 证书。如需更多信息，请参阅*从过期的 control plane 证书中恢复*的文档。
- 建议您在生成 12 小时后使用 Ignition 配置文件，因为集群安装后 24 小时证书从 16 小时轮转至 22 小时。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中运行证书更新时避免安装失败。

先决条件

- 已获得 OpenShift Container Platform 安装程序。
- 已创建 **install-config.yaml** 安装配置文件。

流程

1. 切换到包含安装程序的目录，并为集群生成 Kubernetes 清单：

```
$ ./openshift-install create manifests --dir <installation_directory> 1
```

1 对于 **<installation_directory>**，请指定含有您创建的 **install-config.yaml** 文件的安装目录。

2. 检查 **<installation_directory>/manifests/cluster-scheduler-02-config.yml** Kubernetes 清单文件中的 **mastersSchedulable** 参数是否已设置为 **false**。此设置可防止在 control plane 机器上调度 pod:
   a. 打开 **<installation_directory>/manifests/cluster-scheduler-02-config.yml** 文件。
   b. 找到 **mastersSchedulable** 参数并确保它被设置为 **false**。
   c. 保存并退出文件。
3. 要创建 Ignition 配置文件，从包含安装程序的目录运行以下命令：

```
$ ./openshift-install create ignition-configs --dir <installation_directory> 1
```

1 对于 **<installation_directory>**，请指定相同的安装目录。

该目录中将生成以下文件：

```
.
├── auth
│   ├── kubeadmin-password
│   └── kubeconfig
├── bootstrap.ign
├── master.ign
├── metadata.json
└── worker.ign
```

7.1.11. 创建 Red Hat Enterprise Linux CoreOS (RHCOS) 机器

在您置备的 IBM Z 环境中安装集群前，您必须在 z/VM 虚拟机上安装 RHCOS 以便集群使用。完成以下步骤以创建机器。

先决条件

- 在置备机器中运行 FTP 服务器，您创建的机器需要可以访问这个 FTP 服务器。

流程

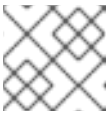
1. 在您置备的机器上登录到 Linux。
2. 从 RHCOS 镜像镜像 获取 Red Hat Enterprise Linux CoreOS（RHCOS）内核、initramfs 和 rootfs 文件。

重要

RHCOS 镜像可能不会随着 OpenShift Container Platform 的每一发行版本都有改变。您必须下载最高版本的镜像，其版本号应小于或等于您安装的 OpenShift Container Platform 版本。只使用以下流程中描述的适当内核、initramfs 和 rootfs 工件。

文件名包含 OpenShift Container Platform 版本号。它们类似以下示例：

- kernel: **rhcos-<version>-live-kernel-<architecture>**
- initramfs: **rhcos-<version>-live-initramfs.<architecture>.img**
- rootfs: **rhcos-<version>-live-rootfs.<architecture>.img**



注意

FCP 和 DASD 的 rootfs 镜像是相同的。

3. 创建参数文件。以下参数特定于特定虚拟机：
   - 对于 **coreos.inst.install_dev=**，请为 DASD 安装指定 **dasda**，或者为 FCP 指定 **sda**。请注意 FCP 需要 **zfcp.allow_lun_scan=0**。
   - 对于 **rd.dasd=**，请指定要安装 RHCOS 的 DASD。
   - **rd.zfcp=<adapter>,<wwpn>,<lun>** 指定要在其中安装 RHCOS 的 FCP 磁盘。
   - 对于 **ip=**，请指定以下七项：
     i. 机器的 IP 地址。
     ii. 一个空字符串。
     iii. 网关
     iv. 子网掩码。

v. **hostname.domainname**格式的机器主机和域名。省略这个值会让 RHCOS 来决定这个值。

vi. 网络接口名称。省略这个值会让 RHCOS 来决定这个值。

vii. 如果使用静态 IP 地址，则为一个空字符串。

- 对于 **coreos.inst.ignition_url=**，为机器角色指定 Ignition 文件。使用 **bootstrap.ign**、**master.ign** 或 **worker.ign**。只支持 HTTP 和 HTTPS 协议。
- 对于 **coreos.live.rootfs_url=**，为您引导的内核和 initramfs 指定匹配的 rootfs 工件。只支持 HTTP 和 HTTPS 协议。
- 所有其他参数都可以保留。
  bootstrap 机器的实例参数文件（**bootstrap-0.parm**）如下：

```
rd.neednet=1 \
console=ttysclp0 \
coreos.inst.install_dev=dasda \
coreos.live.rootfs_url=http://cl1.provide.example.com:8080/assets/rhcos-live-rootfs.s390x.img \
coreos.inst.ignition_url=http://cl1.provide.example.com:8080/ignition/bootstrap.ign \
ip=172.18.78.2::172.18.78.1:255.255.255.0:::none nameserver=172.18.78.1 \
rd.znet=qeth,0.0.bdf0,0.0.bdf1,0.0.bdf2,layer2=1,portno=0 \
zfcp.allow_lun_scan=0 \
rd.dasd=0.0.3490
```

将参数文件中的所有选项写为一行，并确保您没有换行字符。

4. 将 initramfs 、内核、参数文件和 RHCOS 镜像传送到 z/VM 中，例如使用 FTP。有关如何使用 FTP 传输文件并从虚拟 reader 引导的详情，请参考 在 Z/VM 中安装。
5. 将文件 punch 到 z/VM 虚拟机的虚拟 reader，即成为 bootstrap 节点。
   请参阅 IBM 文档中的 PUNCH。

   提示

   您可以使用 CP PUNCH 命令（如果是 Linux，使用 **vmur** 命令）在两个 z/VM 虚拟机间传输文件。

6. 在 bootstrap 机器中登录到 CMS。
7. 从 reader IPL bootstrap 机器：

```
$ ipl c
```

   请参阅 IBM 文档中的 IPL。

8. 对集群中的其他机器重复此步骤。

7.1.11.1. 高级 RHCOS 安装参考

本节演示了网络配置和其他高级选项，允许您修改 Red Hat Enterprise Linux CoreOS（RHCOS）手动安装过程。下表描述了您可以与 RHCOS live installer 和 **coreos-installer** 命令一起使用的内核参数和命令行选项。

RHCOS 启动提示下的路由和绑定选项
如果从 ISO 镜像安装 RHCOS，您可以在引导该镜像时手动添加内核参数以配置节点的网络。如果没有使用网络参数，则安装默认为使用 DHCP。

重要

添加网络参数时，还必须添加 **rd.neednet=1** 内核参数。

下表描述了如何为实时 ISO 安装使用 **ip=** 、 **nameserver=** 和 **bond=** 内核参数。

注意

在添加内核参数时顺序非常重要： **ip=**，**nameserver=**，然后 **bond=**。

ISO 的路由和绑定选项

下表提供了配置 Red Hat Enterprise Linux CoreOS（RHCOS）节点网络的示例。这些是在系统引导过程中传递给 **dracut** 工具的网络选项。有关 **dracut** 支持的网络选项的详情，请参考 **dracut.cmdline** 手册页。

描述	例子
要配置一个 IP 地址，可以使用 DHCP(**ip=dhcp**)或者设置单独的静态 IP 地址(**ip=<host_ip>**)。然后在每个节点上指定 DNS 服务器 IP 地址(**nameserver=<dns_ip>**)。这个示例设置： • 节点的 IP 地址为 **10.10.10.2** • 网关地址为 **10.10.10.254** • 子网掩码为 **255.255.255.0** • 主机名为 **core0.example.com** • DNS 服务器地址为 **4.4.4.41**	ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp1s0:none nameserver=4.4.4.41
通过指定多个 **ip=** 条目来指定多个网络接口。	ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp1s0:none ip=10.10.10.3::10.10.10.254:255.255.255.0:core0.example.com:enp2s0:none
可选：您可以通过设置一个 **rd.route=** 值来配置到额外网络的路由。 如果额外网络网关与主要网络网关不同，则默认网关必须是主要网络网关。	配置默认网关： ip=::10.10.10.254:::: 为额外网络配置路由： rd.route=20.20.20.0/24:20.20.20.254:enp2s0

描述	例子
在单一接口中禁用 DHCP，比如当有两个或者多个网络接口时，且只有一个接口被使用。	ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp1s0:none ip=::::core0.example.com:enp2s0:none
您可以将系统中 DHCP 和静态 IP 配置与多个网络接口结合在一起。	ip=enp1s0:dhcp ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp2s0:none
可选： 您可以使用 **vlan=** 参数在单独的接口上配置 VLAN。	在网络接口中配置 VLAN 并使用静态 IP 地址： ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp2s0.100:none vlan=enp2s0.100:enp2s0 在网络接口中配置 VLAN 并使用 DHCP： ip=enp2s0.100:dhcp vlan=enp2s0.100:enp2s0
您可以为每个服务器添加一个 **nameserver=** 条目来提供多个 DNS 服务器。	nameserver=1.1.1.1 nameserver=8.8.8.8
可选：使用 **bond=** 选项支持将多个网络接口绑定到一个接口。在这两个示例中： • 配置绑定接口的语法为：**bond=name[:network_interfaces][:options]** • *name* 是绑定设备名称（**bond0**），*network_interfaces* 代表用逗号分开的物理（以太网）接口(**em1,em2**)的列表，*options* 是用逗号分开的绑定选项列表。输入 **modinfo bonding** 查看可用选项。 • 当使用 **bond=** 创建绑定接口时，您必须指定如何分配 IP 地址以及绑定接口的其他信息。	要将绑定的接口配置为使用 DHCP，请将绑定的 IP 地址设置为 **dhcp**。例如： bond=bond0:em1,em2:mode=active-backup ip=bond0:dhcp 要将绑定接口配置为使用静态 IP 地址，请输入您需要的特定 IP 地址以及相关信息。例如： bond=bond0:em1,em2:mode=active-backup ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:bond0:none

描述	例子
可选： 您可以使用 **vlan=** 参数在绑定接口上配置 VLAN。	使用 VLAN 配置绑定接口并使用 DHCP： ip=bond0.100:dhcp bond=bond0:em1,em2:mode=active-backup vlan=bond0.100:bond0 使用 VLAN 配置绑定接口，并使用静态 IP 地址： ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:bond0.100:none bond=bond0:em1,em2:mode=active-backup vlan=bond0.100:bond0
可选：通过使用 **team=** 参数，网络合作可用作绑定的替代选择。在此例中： • 配置组接口的语法为：**team=name[:network_interfaces]** *name* 是组设备名称(**team0**)，**network_interfaces** 代表以逗号分隔的物理（以太网）接口（**em1**、**em2**）列表。 **注意** 当 RHCOS 切换到即将推出的 RHEL 版本时，团队计划会被弃用。如需更多信息，请参阅红帽知识库文章。	配置网络团队： team=team0:em1,em2 ip=team0:dhcp

7.1.12. 创建集群

要创建 OpenShift Container Platform 集群，请等待您通过安装程序生成的 Ignition 配置文件所置备的机器上完成 bootstrap 过程。

先决条件

- 为集群创建所需的基础架构。
- 已获得安装程序并为集群生成了 Ignition 配置文件。
- 已使用 Ignition 配置文件为集群创建 RHCOS 机器。
- 您的机器可直接访问互联网，或者可以使用 HTTP 或 HTTPS 代理。

流程

1. 监控 bootstrap 过程：

```
$ ./openshift-install --dir <installation_directory> wait-for bootstrap-complete \ 1
    --log-level=info 2
```

1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

2 要查看不同的安装详情，请指定 **warn**、**debug** 或 **error**，而不要指定 **info**。

输出示例

```
INFO Waiting up to 30m0s for the Kubernetes API at https://api.test.example.com:6443...
INFO API v1.19.0 up
INFO Waiting up to 30m0s for bootstrapping to complete...
INFO It is now safe to remove the bootstrap resources
```

Kubernetes API 服务器提示已在 control plane 机器上完成 bootstrap 时，命令运行成功。

2. bootstrap 过程完成后，请从负载均衡器中删除 bootstrap 机器。

重要

此时您必须从负载均衡器中删除 bootstrap 机器。您还可以删除或重新格式化机器本身。

7.1.13. 使用 CLI 登录到集群

您可以通过导出集群 **kubeconfig** 文件，以默认系统用户身份登录集群。**kubeconfig** 文件包含关于集群的信息，供 CLI 用于将客户端连接到正确集群和 API 服务器。该文件特只适用于一个特定的集群，在 OpenShift Container Platform 安装过程中创建。

先决条件

- 已部署了 OpenShift Container Platform 集群。
- 已安装 **oc** CLI。

流程

1. 导出 **kubeadmin** 凭证：

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

2. 使用导出的配置，验证能否成功运行 **oc** 命令：

```
$ oc whoami
```

输出示例

```
system:admin
```

7.1.14. 批准机器的证书签名请求

将机器添加到集群时，会为您添加的每台机器生成两个待处理证书签名请求（CSR）。您必须确认这些 CSR 已获得批准，或根据需要自行批准。客户端请求必须首先被批准，然后是服务器请求。

先决条件

- 您已将机器添加到集群中。

流程

1. 确认集群可以识别这些机器：

```
$ oc get nodes
```

输出示例

```
NAME      STATUS    ROLES   AGE  VERSION
master-0  Ready     master  63m  v1.19.0
master-1  Ready     master  63m  v1.19.0
master-2  Ready     master  64m  v1.19.0
```

输出将列出您创建的所有机器。

注意

在一些 CSR 被批准前，以上输出可能不包括计算节点（也称为 worker 节点）。

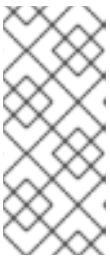
2. 检查待处理的 CSR，并确保可以看到添加到集群中的每台机器都有 **Pending** 或 **Approved** 状态的客户端请求：

```
$ oc get csr
```

输出示例

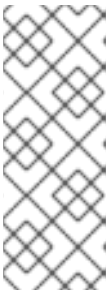
```
NAME        AGE   REQUESTOR                                   CONDITION
csr-mddf5   20m   system:node:master-01.example.com   Approved,Issued
csr-z5rln   16m   system:node:worker-21.example.com   Approved,Issued
```

3. 如果 CSR 没有获得批准，请在所添加机器的所有待处理 CSR 都处于 **Pending** 状态后，为您的集群机器批准这些 CSR：



注意

由于 CSR 会自动轮转，因此请在将机器添加到集群后一小时内批准您的 CSR。如果没有在一小时内批准，证书将会轮转，每个节点将会存在多个证书。您必须批准所有这些证书。批准客户端 CSR 后， Kubelet 为服务证书创建辅助 CSR，这需要手动批准。然后，如果 Kubelet 请求具有相同参数的新证书，则 **machine-approver** 会自动批准后续服务证书续订请求。



注意

对于在未启用机器 API 的平台中运行的集群，如裸机和其他用户置备的基础架构，必须采用一种方法自动批准 kubelet 提供证书请求（CSR）。如果没有批准请求，则 **oc exec**、**oc rsh** 和 **oc logs** 命令将无法成功，因为 API 服务器连接到 kubelet 时需要服务证书。与 Kubelet 端点联系的任何操作都需要此证书批准。这个方法必须监视新的 CSR，确认 CSR 由 **system:node** 或 **system:admin** 组中的 **node-bootstrapper** 服务帐户提交，并确认节点的身份。

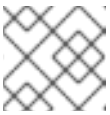
- 若要单独批准，请对每个有效的 CSR 运行以下命令：

```
$ oc adm certificate approve <csr_name> 1
```

1 **<csr_name>** 是当前 CSR 列表中 CSR 的名称。

- 要批准所有待处理的 CSR，请运行以下命令：

```
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}{{"\n"}}
{{end}}{{end}}' | xargs --no-run-if-empty oc adm certificate approve
```



注意

在有些 CSR 被批准前，一些 Operator 可能无法使用。

4. 现在，您的客户端请求已被批准，您必须查看添加到集群中的每台机器的服务器请求：

```
$ oc get csr
```

输出示例

```
NAME        AGE     REQUESTOR                                                   CONDITION
csr-bfd72   5m26s   system:node:ip-10-0-50-126.us-east-2.compute.internal
Pending
csr-c57lv   5m26s   system:node:ip-10-0-95-157.us-east-2.compute.internal
Pending
...
```

5. 如果剩余的 CSR 没有被批准，且处于 **Pending** 状态，请批准集群机器的 CSR：

- 若要单独批准，请对每个有效的 CSR 运行以下命令：

```
$ oc adm certificate approve <csr_name> 1
```

1 **<csr_name>** 是当前 CSR 列表中 CSR 的名称。

- 要批准所有待处理的 CSR，请运行以下命令：

```
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}{{"\n"}}
{{end}}{{end}}' | xargs oc adm certificate approve
```

6. 批准所有客户端和服务器 CSR 后，器将处于 **Ready** 状态。运行以下命令验证：

```
$ oc get nodes
```

输出示例

```
NAME     STATUS   ROLES   AGE  VERSION
master-0 Ready    master  73m  v1.20.0
master-1 Ready    master  73m  v1.20.0
master-2 Ready    master  74m  v1.20.0
worker-0 Ready    worker  11m  v1.20.0
worker-1 Ready    worker  11m  v1.20.0
```

注意

批准服务器 CSR 后可能需要几分钟时间让机器转换为 **Ready** 状态。

其他信息

- 如需有关 CSR 的更多信息，请参阅证书签名请求。

7.1.15. 初始 Operator 配置

在 control plane 初始化后，您必须立即配置一些 Operator 以便它们都可用。

先决条件

- 您的 control plane 已初始化。

流程

1. 观察集群组件上线：

```
$ watch -n5 oc get clusteroperators
```

输出示例

```
NAME                                       VERSION AVAILABLE   PROGRESSING   DEGRADED
SINCE
authentication                             4.6.0   True        False         False      3h56m
cloud-credential                           4.6.0   True        False         False      29h
cluster-autoscaler                         4.6.0   True        False         False      29h
config-operator                            4.6.0   True        False         False      6h39m
console                                    4.6.0   True        False         False      3h59m
csi-snapshot-controller                    4.6.0   True        False         False      4h12m
dns                                        4.6.0   True        False         False      4h15m
etcd                                       4.6.0   True        False         False      29h
image-registry                             4.6.0   True        False         False      3h59m
ingress                                    4.6.0   True        False         False      4h30m
insights                                   4.6.0   True        False         False      29h
kube-apiserver                             4.6.0   True        False         False      29h
kube-controller-manager                    4.6.0   True        False         False      29h
kube-scheduler                             4.6.0   True        False         False      29h
kube-storage-version-migrator              4.6.0   True        False         False      4h2m
machine-api                                4.6.0   True        False         False      29h
```

```
machine-approver                           4.6.0   True        False       False      6h34m
machine-config                             4.6.0   True        False       False      3h56m
marketplace                                4.6.0   True        False       False      4h2m
monitoring                                 4.6.0   True        False       False      6h31m
network                                    4.6.0   True        False       False      29h
node-tuning                                4.6.0   True        False       False      4h30m
openshift-apiserver                        4.6.0   True        False       False      3h56m
openshift-controller-manager               4.6.0   True        False       False      4h36m
openshift-samples                          4.6.0   True        False       False      4h30m
operator-lifecycle-manager                 4.6.0   True        False       False      29h
operator-lifecycle-manager-catalog         4.6.0   True        False       False      29h
operator-lifecycle-manager-packageserver   4.6.0   True        False       False      3h59m
service-ca                                 4.6.0   True        False       False      29h
storage                                    4.6.0   True        False       False      4h30m
```

2. 配置不可用的 Operator。

7.1.15.1. 镜像 registry 存储配置

对于不提供默认存储的平台，Image Registry Operator 最初将不可用。安装后，您必须配置 registry 使用的存储，这样 Registry Operator 才可用。

示配置生产集群所需的持久性卷的说明。如果适用，显示有关将空目录配置为存储位置的说明，该位置只可用于非生产集群。

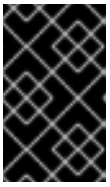
另外还提供了在升级过程中使用 **Recreate** rollout 策略来允许镜像 registry 使用块存储类型的说明。

7.1.15.1.1. 为 IBM Z 配置 registry 存储

作为集群管理员，在安装后需要配置 registry 来使用存储。

先决条件

- 具有 Cluster Administrator 权限
- IBM Z 上的集群。
- 为集群置备的持久性存储。



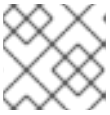
重要

如果您只有一个副本，OpenShift Container Platform 支持对镜像 registry 存储的 **ReadWriteOnce** 访问。要部署支持高可用性的、带有两个或多个副本的镜像 registry，需要 **ReadWriteMany** 访问设置。

- 必须具有 100Gi 容量。

流程

1. 为了配置 registry 使用存储，需要修改 **configs.imageregistry/cluster** 资源中的 **spec.storage.pvc** 。



注意

使用共享存储时，请查看您的安全设置以防止被外部访问。

2. 验证您没有 registry pod:

```
$ oc get pod -n openshift-image-registry
```

注意

如果存储类型为 **emptyDIR**，则副本数不能超过 **1**。

3. 检查 registry 配置：

```
$ oc edit configs.imageregistry.operator.openshift.io
```

输出示例

```
storage:
  pvc:
    claim:
```

将 **claim** 字段留空以允许自动创建一个 **image-registry-storage** PVC。

4. 检查 **clusteroperator** 的状态：

```
$ oc get clusteroperator image-registry
```

5. 确保您的 registry 设置为 manage，以启用镜像的构建和推送。

- 运行：

```
$ oc edit configs.imageregistry/cluster
```

然后将行改

```
managementState: Removed
```

为

```
managementState: Managed
```

7.1.15.1.2. 在非生产集群中配置镜像 registry 存储

您必须为 Image Registry Operator 配置存储。对于非生产集群，您可以将镜像 registry 设置为空目录。如果您这样做，重启 registry 后会丢失所有镜像。

流程

- 将镜像 registry 存储设置为空目录：

```
$ oc patch configs.imageregistry.operator.openshift.io cluster --type merge --patch '{"spec":{"storage":{"emptyDir":{}}}}'
```

警告

仅可为非生产集群配置这个选项。

如果在 Image Registry Operator 初始化其组件前运行此命令，**oc patch** 命令会失败并显示以下错误：

```
Error from server (NotFound): configs.imageregistry.operator.openshift.io "cluster" not found
```

等待几分钟，然后再次运行该命令。

7.1.16. 在用户置备的基础架构上完成安装

完成 Operator 配置后，可以在您提供的基础架构上完成集群安装。

先决条件

- 您的 control plane 已初始化。
- 已完成初始 Operator 配置。

流程

1. 使用以下命令确认所有集群组件都已在线：

```
$ watch -n5 oc get clusteroperators
```

输出示例

```
NAME                                       VERSION AVAILABLE   PROGRESSING   DEGRADED
SINCE
authentication                             4.6.0   True        False         False      3h56m
cloud-credential                           4.6.0   True        False         False      29h
cluster-autoscaler                         4.6.0   True        False         False      29h
config-operator                            4.6.0   True        False         False      6h39m
console                                    4.6.0   True        False         False      3h59m
csi-snapshot-controller                    4.6.0   True        False         False      4h12m
dns                                        4.6.0   True        False         False      4h15m
etcd                                       4.6.0   True        False         False      29h
image-registry                             4.6.0   True        False         False      3h59m
ingress                                    4.6.0   True        False         False      4h30m
insights                                   4.6.0   True        False         False      29h
kube-apiserver                             4.6.0   True        False         False      29h
kube-controller-manager                    4.6.0   True        False         False      29h
kube-scheduler                             4.6.0   True        False         False      29h
```

```
kube-storage-version-migrator              4.6.0   True        False         False      4h2m
machine-api                                4.6.0   True        False         False      29h
machine-approver                           4.6.0   True        False         False      6h34m
machine-config                             4.6.0   True        False         False      3h56m
marketplace                                4.6.0   True        False         False      4h2m
monitoring                                 4.6.0   True        False         False      6h31m
network                                    4.6.0   True        False         False      29h
node-tuning                                4.6.0   True        False         False      4h30m
openshift-apiserver                        4.6.0   True        False         False      3h56m
openshift-controller-manager               4.6.0   True        False         False      4h36m
openshift-samples                          4.6.0   True        False         False      4h30m
operator-lifecycle-manager                 4.6.0   True        False         False      29h
operator-lifecycle-manager-catalog         4.6.0   True        False         False      29h
operator-lifecycle-manager-packageserver   4.6.0   True        False         False      3h59m
service-ca                                 4.6.0   True        False         False      29h
storage                                    4.6.0   True        False         False      4h30m
```

或者，通过以下命令，如果所有集群都可用您会接到通知。它还检索并显示凭证：

```
$ ./openshift-install --dir <installation_directory> wait-for install-complete 1
```

1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

输出示例

```
INFO Waiting up to 30m0s for the cluster to initialize...
```

Cluster Version Operator 完成从 Kubernetes API 服务器部署 OpenShift Container Platform 集群时，命令运行成功。

重要

- 安装程序生成的 Ignition 配置文件包含在 24 小时后过期的证书，然后在过期时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外情况是，您需要手动批准待处理的 **node-bootstrapper** 证书签名请求（CSR）来恢复 kubelet 证书。如需更多信息，请参阅*从过期的 control plane 证书中恢复*的文档。
- 建议您在生成 12 小时后使用 Ignition 配置文件，因为集群安装后 24 小时证书从 16 小时轮转至 22 小时。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中运行证书更新时避免安装失败。

2. 确认 Kubernetes API 服务器正在与 pod 通信。

   a. 要查看所有 pod 的列表，请使用以下命令：

   ```
   $ oc get pods --all-namespaces
   ```

   输出示例

   ```
   NAMESPACE                       NAME                                         READY   STATUS
   RESTARTS   AGE
   openshift-apiserver-operator    openshift-apiserver-operator-85cb746d55-zqhs8   1/1
   ```

```
Running    1          9m
openshift-apiserver            apiserver-67b9g                          1/1     Running     0
3m
openshift-apiserver            apiserver-ljcmx                          1/1     Running     0
1m
openshift-apiserver            apiserver-z25h4                          1/1     Running     0
2m
openshift-authentication-operator authentication-operator-69d5d8bf84-vh2n8        1/1
Running    0          5m
...
```

b. 使用以下命令，查看上一命令的输出中所列 pod 的日志：

```
$ oc logs <pod_name> -n <namespace> 1
```

1 指定 pod 名称和命名空间，如上一命令的输出中所示。

如果 pod 日志显示，Kubernetes API 服务器可以与集群机器通信。

7.1.17. OpenShift Container Platform 的 Telemetry 访问

在 OpenShift Container Platform 4.6 中，默认运行的 Telemetry 服务提供有关集群健康状况和成功更新的指标，需要访问互联网。如果您的集群连接到互联网，Telemetry 会自动运行，而且集群会注册到 OpenShift Cluster Manager。

确认 OpenShift Cluster Manager 清单正确后，可以由 Telemetry 自动维护，也可以使用 OpenShift Cluster Manager 手动维护，使用订阅监控来跟踪帐户或多集群级别的 OpenShift Container Platform 订阅。

其他资源

- 有关 Telemetry 服务的更多信息，请参阅关于远程健康监控。

7.1.18. 收集调试信息

您可以收集有助于在 IBM Z 中安装 OpenShift Container Platform 时对特定问题进行故障排除和调试的调试信息。

先决条件

- 安装了 **oc** CLI 工具

流程

1. 登录到集群：

```
$ oc login
```

2. 在您要收集硬件信息的节点中，启动一个调试容器：

```
$ oc debug node/<nodename>
```

3. 进入 **/host** 文件系统并启动 **toolbox**:

```
$ chroot /host
$ toolbox
```

4. 收集 **dbginfo** 数据：

```
$ dbginfo.sh
```

5. 然后可以使用 **scp**来获取数据。

其他资源

- 请参阅 如何在没有 SSH 的 OpenShift4 节点中生成 SOSREPORT 。

7.1.19. 后续步骤

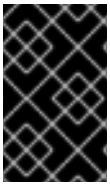
- 自定义集群。
- 如果需要，您可以选择不使用远程健康报告。

7.2. 在受限网络中在 IBM Z 和 LINUXONE 上安装集群

在 OpenShift Container Platform 版本 4.6 中，可以在受限网络中置备的 IBM Z 和 LinuxONE 基础架构上安装集群。

注意

虽然本文档仅涉及了 IBM Z，但它的所有信息也适用于 LinuxONE。

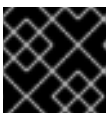


重要

非裸机平台还有其他注意事项。在尝试在此类环境中安装 OpenShift Container Platform 集群前，请参阅有关在未经测试的平台上部署 OpenShift Container Platform 的指南 中的信息。

先决条件

- 在受限网络中创建镜像 registry，并获取您的 OpenShift Container Platform 版本的 **imageContentSources** 数据。
- 在开始安装前，必须移动或删除现有的安装文件。这可保证在安装过程中创建和更新所需的安装文件。



重要

确定从可访问安装介质的机器中执行安装步骤。

- 为集群置备使用 NFS 的持久性存储。若要部署私有镜像 registry，您的存储必须提供 **ReadWriteMany** 访问模式。
- 查看有关 OpenShift Container Platform 安装和更新 流程的详细信息。
- 如果使用防火墙并计划使用遥测（telemetry），您必须将防火墙配置为允许集群需要访问的站点。

注意

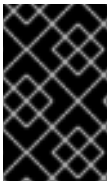
如果您要配置代理，请务必也要查看此站点列表。

7.2.1. 关于在受限网络中安装

在 OpenShift Container Platform 4.6 中，可以执行不需要有效的互联网连接来获取软件组件的安装。受限网络安装可使用安装程序置备的基础架构或用户置备的基础架构完成，具体取决于您要安装集群的云平台。

如果选择在云平台中执行受限网络安装，仍然需要访问其云 API。有些云功能，比如 Amazon Web Service 的 Route 53 DNS 和 IAM 服务，需要访问互联网。根据您的网络，在裸机硬件或 VMware vSphere 上安装时可能需要较少的互联网访问。

要完成受限网络安装，您必须创建一个 registry，镜像 OpenShift Container Platform registry 的内容并包含其安装介质。您可以在堡垒主机上创建此镜像，该主机可同时访问互联网和您的封闭网络，也可以使用满足您的限制条件的其他方法。



重要

由于用户置备安装配置的复杂性，在尝试使用用户置备的基础架构受限网络安装前，请考虑完成标准用户置备的基础架构安装。通过完成此测试安装，您可以更轻松地隔离和排查您在受限网络中安装时可能出现的问题。

7.2.1.1. 其他限制

受限网络中的集群还有以下额外限制：

- **ClusterVersion** 状态包含一个 **Unable to retrieve available updates** 错误。
- 默认情况下，您无法使用 Developer Catalog 的内容，因为您无法访问所需的镜像流标签。

7.2.2. OpenShift Container Platform 的互联网访问

在 OpenShift Container Platform 4.6 中，您需要访问互联网来获得用来安装集群的镜像。

您必须具有以下互联网访问权限：

- 访问 OpenShift Cluster Manager 以下载安装程序并执行订阅管理。如果集群可以访问互联网，并且没有禁用 Telemetry，该服务会自动授权您的集群。
- 访问 Quay.io，以获取安装集群所需的软件包。
- 获取执行集群更新所需的软件包。

重要

如果您的集群无法直接访问互联网，则可以在置备的某些类基础架构上执行受限网络安装。在此过程中，您要下载所需的内容，并使用它在镜像 registry（mirror registry） 中填充安装集群并生成安装程序所需的软件包。对于某些安装类型，集群要安装到的环境不需要访问互联网。在更新集群之前，要更新 registry 镜像系统中的内容。

7.2.3. 具有用户置备基础架构的集群的机器要求

对于含有用户置备的基础架构的集群，您必须部署所有所需的机器。

7.2.3.1. 所需的机器

最小的 OpenShift Container Platform 集群需要下列主机：

- 一个临时 bootstrap 机器
- 三台 control plane 或 master 机器
- 至少两台计算机器，也称为 worker 机器。

注意

集群要求 bootstrap 机器在三台 control plane 机器上部署 OpenShift Container Platform 集群。您可在安装集群后删除 bootstrap 机器。

重要

要提高集群的高可用性，请在最少两个物理集群中的不同的 z/VM 实例中分布运行 control plane 机器。

bootstrap 和 control plane 机器必须使用 Red Hat Enterprise Linux CoreOS (RHCOS) 作为操作系统。但是，计算机器可以在 Red Hat Enterprise Linux CoreOS(RHCOS)或 Red Hat Enterprise Linux(RHEL)7.9 之间进行选择。

请注意，RHCOS 基于 Red Hat Enterprise Linux（RHEL） 8，并继承其所有硬件认证和要求。请查看Red Hat Enterprise Linux 技术功能及限制 。

7.2.3.2. 网络连接要求

所有 Red Hat Enterprise Linux CoreOS (RHCOS) 机器在启动过程中需要 **initramfs** 中的网络从 Machine Config Server 获取 Ignition 配置文件。机器被配置为使用静态 IP 地址。不需要 DHCP 服务器。另外，集群中的每个 OpenShift Container Platform 节点都必须有权访问网络时间协议（NTP）服务器。

7.2.3.3. IBM Z 网络连接要求

要在 z/VM 中安装 IBM Z，您需要使用第 2 层模式的单一 z/VM 虚拟 NIC。您还需要：

- 直接连接的 OSA 或 RoCE 网络适配器
- z/VM vSwitch 设置。对于首选的设置，请使用 OSA 链接聚合。

7.2.3.4. 最低资源要求

每台集群机器都必须满足以下最低要求：

表 7.11. 最低资源要求

机器	操作系统	vCPU [1]	虚拟内存	存储	IOPS
bootstrap	RHCOS	4	16 GB	100 GB	N/A
Control plane	RHCOS	4	16 GB	100 GB	N/A

机器	操作系统	vCPU [1]	虚拟内存	存储	IOPS
Compute	RHCOS	2	8 GB	100 GB	N/A

1. 当未启用并发多线程 (SMT) 或超线程时，一个 vCPU 相当于一个物理内核。启用后，使用以下公式来计算对应的比例：（每个内核数的线程）× sockets = vCPU。

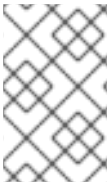
7.2.3.5. 最低 IBM Z 系统环境

您可以在以下 IBM 硬件上安装 OpenShift Container Platform 版本 4.6：

- IBM z15（所有型号）、IBM z14（所有型号）、IBM z13 和 IBM z13s
- 任何版本的 LinuxONE

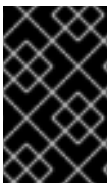
硬件要求

- 为每个集群启用 SMT2 相当于 6 个 IFL。
- 至少一个网络连接连接到 **LoadBalancer** 服务，并为集群外的流量提供数据。



注意

您可以使用专用或共享的 IFL 来分配足够的计算资源。资源共享是 IBM Z 的一个关键优势。但是，您必须正确调整每个虚拟机监控程序层上的容量，并确保每个 OpenShift Container Platform 集群都有充足的资源。



重要

由于集群的整体性能可能会受到影响，用于设置 OpenShift Container Platform 集群的 LPAR 必须提供足够的计算容量。就此而言，管理程序级别上的 LPAR 权重管理、授权和 CPU 共享扮演着重要角色。

操作系统要求

- 一个 z/VM 7.1 或更高版本的实例

在您的 z/VM 实例中设置：

- 3 个客户虚拟机作为 OpenShift Container Platform control plane 的机器
- 2 个客户虚拟机作为 OpenShift Container Platform 的计算机器
- 1 个客户虚拟机作为临时 OpenShift Container Platform bootstrap 机器

IBM Z 网络连接要求
要在 z/VM 中安装 IBM Z，您需要使用第 2 层模式的单一 z/VM 虚拟 NIC。您还需要：

- 直接连接的 OSA 或 RoCE 网络适配器
- z/VM vSwitch 设置。对于首选的设置，请使用 OSA 链接聚合。

z/VM 客户虚拟机的磁盘存储

- 附加了 FICON 的磁盘存储。可以是 z/VM Minidisks、fullpack Minidisks 或专用 DASD，它们都必须被格式化为 CDL，这是默认的 CDL。要达到 Red Hat Enterprise Linux CoreOS (RHCOS) 安装所需的最小 DASD 大小，您需要扩展地址卷 (EAV)。如果可用，使用 HyperPAV 来确保最佳性能。
- FCP 连接的磁盘存储

存储/主内存

- OpenShift Container Platform control plane 需要 16 GB
- OpenShift Container Platform 计算机器需要 8 GB
- 临时 OpenShift Container Platform bootstrap 机器需要 16 GB

7.2.3.6. 首选 IBM Z 系统环境

硬件要求

- 3 个 LPARS，每个集群都有相当于 6 个 IFL 的 LPARS，它们启用了 SMT2。
- 两个网络连接连接到 **LoadBalancer** 服务，并为集群外的流量提供数据。
- HiperSockets，可以直接作为设备附加到节点，或者与一个 z/VM VSWITCH 桥接，对 z/VM 客户机透明。要将 HiperSockets 直接连接到节点，您必须通过 RHEL 8 客户机设置到外部网络的网关来桥接到 HiperSockets 网络。

操作系统要求

- 2 个或 3 个 z/VM 7.1 或更高版本的实例以实现高可用性

在您的 z/VM 实例中设置：

- 3 个用于 OpenShift Container Platform control plane 机器的虚拟机，每个 z/VM 实例一个。
- 至少 6 个用于 OpenShift Container Platform 计算机器的虚拟机，分布在 z/VM 实例中。
- 1 个客户虚拟机作为临时 OpenShift Container Platform bootstrap 机器。
- 要确保过量使用环境中组件的可用性，请使用 CP 命令 **SET SHARE** 来提高 control plane 的优先级。如果存在基础架构节点，则对它们执行相同的操作。请参阅 IBM 文档中的 SET SHARE。

IBM Z 网络连接要求
要在 z/VM 中安装 IBM Z，您需要使用第 2 层模式的单一 z/VM 虚拟 NIC。您还需要：

- 直接连接的 OSA 或 RoCE 网络适配器
- z/VM vSwitch 设置。对于首选的设置，请使用 OSA 链接聚合。

z/VM 客户虚拟机的磁盘存储

- 附加了 FICON 的磁盘存储。可以是 z/VM Minidisks、fullpack Minidisks 或专用 DASD，它们都必须被格式化为 CDL，这是默认的 CDL。要达到 Red Hat Enterprise Linux CoreOS (RHCOS) 安装所需的最小 DASD 大小，您需要扩展地址卷 (EAV)。如果可用，请使用 HyperPAV 和 High Performance FICON（zHPF）来确保最佳性能。
- FCP 连接的磁盘存储

存储/主内存

- OpenShift Container Platform control plane 需要 16 GB
- OpenShift Container Platform 计算机器需要 8 GB
- 临时 OpenShift Container Platform bootstrap 机器需要 16 GB

7.2.3.7. 证书签名请求管理

在使用您置备的基础架构时，集群只能有限地访问自动机器管理，因此您必须提供一种在安装后批准集群证书签名请求 (CSR) 的机制。**kube-controller-manager** 只能批准 kubelet 客户端 CSR。 **machine-approver** 无法保证使用 kubelet 凭证请求的提供证书的有效性，因为它不能确认是正确的机器发出了该请求。您必须决定并实施一种方法，以验证 kubelet 提供证书请求的有效性并进行批准。

其他资源

- 请参阅 IBM 文档中的使用 z/VM 虚拟交换机桥接 HiperSockets LAN。
- 请参阅在 z/VM 的 Linux 客户端中扩展 HyperPAV 别名设备 以获得性能优化。
- 有关 LPAR 权重管理和权利的信息，请参阅 LPAR 性能的主题。

7.2.4. 创建用户置备的基础架构

在部署采用用户置备的基础架构的 OpenShift Container Platform 集群前，您必须创建底层基础架构。

先决条件

- 在为集群创建支持基础架构之前，请参阅OpenShift Container Platform 4.x Tested Integrations页。

流程

1. 在每个节点上配置 DHCP 或设置静态 IP 地址。
2. 提供所需的负载均衡器。
3. 配置机器的端口。
4. 配置 DNS。
5. 确保网络可以正常工作。

7.2.4.1. 用户置备的基础架构对网络的要求

所有 Red Hat Enterprise Linux CoreOS（RHCOS）机器在启动过程中需要 **initramfs** 中的网络从机器配置服务器获取 Ignition 配置。

在初次启动过程中，需要一个 DHCP 服务器或集群中的每个机器都设置了静态 IP 地址来建立网络连接，以下载它们的 Ignition 配置文件。

建议您使用 DHCP 服务器为集群进行长期机器管理。确保 DHCP 服务器已配置为向集群机器提供持久 IP 地址和主机名。

Kubernetes API 服务器必须能够解析集群机器的节点名称。如果 API 服务器和 worker 节点位于不同的区域中，您可以配置默认 DNS 搜索区域，以便 API 服务器能够解析节点名称。另一种支持的方法是始终在节点对象和所有 DNS 请求中使用完全限定域名来指代主机。

您必须配置机器间的网络连接，以便集群组件进行通信。每台机器都必须能够解析集群中所有其他机器的主机名。

表 7.12. 所有机器到所有机器

协议	端口	描述
ICMP	N/A	网络可访问性测试
TCP	**1936**	指标
	9000-**9999**	主机级别的服务，包括端口 **9100**-**9101** 上的节点导出器和端口 **9099** 上的 Cluster Version Operator。
	10250-**10259**	Kubernetes 保留的默认端口
	10256	openshift-sdn
UDP	**4789**	VXLAN 和 Geneve
	6081	VXLAN 和 Geneve
	9000-**9999**	主机级别的服务，包括端口 **9100**-**9101** 上的节点导出器。
TCP/UDP	**30000**-**32767**	Kubernetes 节点端口

表 7.13. 要通过控制平面的所有机器

协议	端口	描述
TCP	**6443**	Kubernetes API

表 7.14. control plane 机器到 control plane 机器

协议	端口	描述
TCP	**2379**-**2380**	etcd 服务器和对等端口

网络拓扑要求
您为集群置备的基础架构必须满足下列网络拓扑要求。

负载均衡器
在安装 OpenShift Container Platform 前，您必须置备两个满足以下要求的负载均衡器：

1. **API 负载均衡器**：提供一个通用端点，供用户（包括人和机器）与平台交互和配置。配置以下条

1. **API 负载均衡器**：提供一个通用端点，供用户（包括人和机器）与平台交互和配置。配置以下条件：

   - 只适用于第 4 层负载均衡。这可被称为 Raw TCP、SSL Passthrough 或者 SSL 桥接模式。如果使用 SSL Bridge 模式，必须为 API 路由启用 Server Name Indication（SNI）。
   - 无状态负载平衡算法。这些选项根据负载均衡器的实现而有所不同。

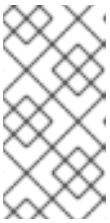
重要

不要为 API 负载均衡器配置会话持久性。

在负载均衡器的前端和后台配置以下端口：

表 7.15. API 负载均衡器

端口	后端机器（池成员）	内部	外部	描述
6443	Bootstrap 和 control plane.bootstrap 机器初始化集群 control plane 后，您要从负载均衡器中删除 bootstrap 机器。您必须为 API 服务器健康检查探测配置 **/readyz** 端点。	X	X	Kubernetes API 服务器
22623	Bootstrap 和 control plane.bootstrap 机器初始化集群 control plane 后，您要从负载均衡器中删除 bootstrap 机器。	X		机器配置服务器



注意

负载均衡器必须配置为，从 API 服务器关闭 **/readyz** 端点到从池中删除 API 服务器实例时最多需要 30 秒。在 **/readyz** 返回错误或处于健康状态后的时间范围内，端点必须被删除或添加。每 5 秒或 10 秒探测一次，有两个成功请求处于健康状态，三个成为不健康的请求经过测试。

2. **应用程序入口负载均衡器**:提供来自集群外部的应用程序流量流量的 Ingress 点。配置以下条件：

   - 只适用于第 4 层负载均衡。这可被称为 Raw TCP、SSL Passthrough 或者 SSL 桥接模式。如果使用 SSL Bridge 模式，您必须为 Ingress 路由启用Server Name Indication（SNI）。
   - 建议根据可用选项以及平台上托管的应用程序类型，使用基于连接的或者基于会话的持久性。

在负载均衡器的前端和后台配置以下端口：

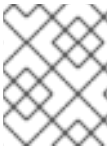
表 7.16. 应用程序入口负载均衡器

端口	后端机器（池成员）	内部	外部	描述
443	默认运行入口路由器 Pod、计算或 worker 的机器。	X	X	HTTPS 流量

端口	后端机器（池成员）	内部	外部	描述
80	默认运行入口路由器 Pod、计算或 worker 的机器。	X	X	HTTP 流量

提示

如果负载均衡器可以看到客户端的真实 IP 地址，启用基于 IP 的会话持久性可提高使用端到端 TLS 加密的应用程序的性能。



注意

OpenShift Container Platform 集群需要正确配置入口路由器。control plane 初始化后，您必须配置入口路由器。

NTP 配置
OpenShift Container Platform 集群默认配置为使用公共网络时间协议（NTP）服务器。如果要使用本地企业 NTP 服务器，或者集群部署在断开连接的网络中，您可以将集群配置为使用特定的时间服务器。如需更多信息，请参阅*配置 chrony 时间服务*的文档。

如果 DHCP 服务器提供 NTP 服务器信息，Red Hat Enterprise Linux CoreOS（RHCOS）机器上的 chrony 时间服务会读取信息，并可与 NTP 服务器同步时钟。:!restricted:

其他资源

- 配置 chrony 时间服务

7.2.4.2. 用户置备 DNS 要求

DNS 用于名称解析和反向名称解析。DNS A/AAAA 或 CNAME 记录用于名称解析，PTR 记录用于反向解析名称。反向记录很重要，因为 Red Hat Enterprise Linux CoreOS（RHCOS）使用反向记录为所有节点设置主机名。另外，反向记录用于生成 OpenShift Container Platform 需要操作的证书签名请求（CSR）。

采用用户置备的基础架构的 OpenShift Container Platform 集群需要以下 DNS 记录。在每一记录中，**<cluster_name>** 是集群名称，**<base_domain>** 则是您在 **install-config.yaml** 文件中指定的集群基域。完整的 DNS 记录采用如下格式: **<component>.<cluster_name>.<base_domain>.**。

表 7.17. 所需的 DNS 记录

组件	记录	描述
Kubernetes API	**api.<cluster_name>.<base_domain>.**	添加 DNS A/AAAA 或 CNAME 记录，以及 DNS PTR 记录，以识别 control plane 机器的负载均衡器。这些记录必须由集群外的客户端以及集群中的所有节点解析。

组件	记录	描述
	api-int.<cluster_name>. <base_domain>.	添加 DNS A/AAAA 或 CNAME 记录，以及 DNS PTR 记录，以识别 control plane 机器的负载均衡器。这些记录必须可以从集群中的所有节点解析。  **重要** API 服务器必须能够根据在 Kubernetes 中记录的主机名解析 worker 节点。如果 API 服务器无法解析节点名称，则代理的 API 调用会失败，且您无法从 pod 检索日志。
Routes	***.apps.<cluster_name>. <base_domain>.**	添加通配符 DNS A/AAAA 或 CNAME 记录，指向以运行入口路由器 Pod 的机器（默认为 worker 节点）为目标的负载均衡器。这些记录必须由集群外的客户端以及集群中的所有节点解析。
bootstrap	**bootstrap.<cluster_name>. <base_domain>.**	添加 DNS A/AAAA 或 CNAME 记录，以及 DNS PTR 记录来识别 bootstrap 机器。这些记录必须由集群中的节点解析。
Master 主机	**<master><n>. <cluster_name>. <base_domain>.**	DNS A/AAAA 或 CNAME 记录，以识别 control plane 节点（也称为 master 节点）的每台机器。这些记录必须由集群中的节点解析。
Worker 主机	**<worker><n>. <cluster_name>. <base_domain>.**	添加 DNS A/AAAA 或 CNAME 记录，以识别 worker 节点的每台机器。这些记录必须由集群中的节点解析。

提示

您可以使用 **nslookup <hostname>** 命令来验证名称解析。您可以使用 **dig -x <ip_address>** 命令来验证 PTR 记录的反向名称解析。

下面的 BIND 区文件的例子展示了关于名字解析的 A 记录的例子。这个示例的目的是显示所需的记录。这个示例不是为选择一个名称解析服务提供建议。

例 7.3. DNS 区数据库示例

```
$TTL 1W
@ IN SOA ns1.example.com. root (
   2019070700 ; serial
   3H  ; refresh (3 hours)
   30M  ; retry (30 minutes)
   2W  ; expiry (2 weeks)
   1W )  ; minimum (1 week)
 IN NS ns1.example.com.
 IN MX 10 smtp.example.com.
```

```
;
;
ns1 IN A 192.168.1.5
smtp IN A 192.168.1.5
;
helper IN A 192.168.1.5
helper.ocp4 IN A 192.168.1.5
;
; The api identifies the IP of your load balancer.
api.ocp4  IN A 192.168.1.5
api-int.ocp4  IN A 192.168.1.5
;
; The wildcard also identifies the load balancer.
*.apps.ocp4  IN A 192.168.1.5
;
; Create an entry for the bootstrap host.
bootstrap.ocp4 IN A 192.168.1.96
;
; Create entries for the master hosts.
master0.ocp4  IN A 192.168.1.97
master1.ocp4  IN A 192.168.1.98
master2.ocp4  IN A 192.168.1.99
;
; Create entries for the worker hosts.
worker0.ocp4  IN A 192.168.1.11
worker1.ocp4  IN A 192.168.1.7
;
;EOF
```

下面的 BIND 区文件示例显示了反向名字解析的 PTR 记录示例。

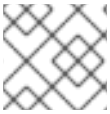
例 7.4. 反向记录的 DNS 区数据库示例

```
$TTL 1W
@ IN SOA ns1.example.com. root (
   2019070700 ; serial
   3H  ; refresh (3 hours)
   30M  ; retry (30 minutes)
   2W  ; expiry (2 weeks)
   1W )  ; minimum (1 week)
 IN NS ns1.example.com.
;
; The syntax is "last octet" and the host must have an FQDN
; with a trailing dot.
97 IN PTR master0.ocp4.example.com.
98 IN PTR master1.ocp4.example.com.
99 IN PTR master2.ocp4.example.com.
;
96 IN PTR bootstrap.ocp4.example.com.
;
5 IN PTR api.ocp4.example.com.
5 IN PTR api-int.ocp4.example.com.
;
11 IN PTR worker0.ocp4.example.com.
```

```
7 IN PTR worker1.ocp4.example.com.
;
;EOF
```

7.2.5. 生成 SSH 私钥并将其添加到代理中

如果要在集群上执行安装调试或灾难恢复，则必须为 **ssh-agent** 和安装程序提供 SSH 密钥。您可以使用此密钥访问公共集群中的 bootstrap 机器来排除安装问题。



注意

在生产环境中，您需要进行灾难恢复和调试。

重要

不要在生产环境中跳过这个过程，因为生产环境需要灾难恢复和调试。

您可以使用此密钥以 **core** 用户身份通过 SSH 连接到 master 节点。在部署集群时，此密钥会添加到 **core** 用户的 **~/.ssh/authorized_keys** 列表中。

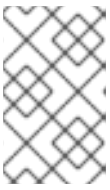
流程

1. 如果还没有为计算机上免密码身份验证而配置的 SSH 密钥，请创建一个。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ ssh-keygen -t ed25519 -N '' \
  -f <path>/<file_name> 1
```

1 指定新 SSH 密钥的路径和文件名，如 **~/.ssh/id_rsa**。如果您已有密钥对，请确保您的公钥位于 **~/.ssh** 目录中。

运行此命令会在指定的位置生成不需要密码的 SSH 密钥。



注意

如果您计划在 **x86_64** 架构中安装使用 FIPS 验证的/Modules in Process 加密库的 OpenShift Container Platform 集群，不要创建使用 **ed25519** 算法的密钥。反之，创建一个使用 **rsa** 或 **ecdsa** 算法的密钥。

2. 作为后台任务启动 **ssh-agent** 进程：

```
$ eval "$(ssh-agent -s)"
```

输出示例

```
Agent pid 31874
```

注意

如果您的集群采用 FIPS 模式，则只使用 FIPS 兼容算法来生成 SSH 密钥。密钥必须是 RSA 或 ECDSA。

3. 将 SSH 私钥添加到 **ssh-agent**：

```
$ ssh-add <path>/<file_name> 1
```

输出示例

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

1 指定 SSH 私钥的路径和文件名，如 **~/.ssh/id_rsa**

后续步骤

- 在安装 OpenShift Container Platform 时，为安装程序提供 SSH 公钥。

7.2.6. 手动创建安装配置文件

对于使用用户置备的基础架构的 OpenShift Container Platform 安装，您必须手动生成安装配置文件。

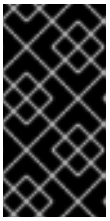
先决条件

- 获取 OpenShift Container Platform 安装程序和集群的访问令牌。
- 获取命令输出中的 **imageContentSources** 部分来镜像存储库。
- 获取您的镜像 registry 的证书内容。

流程

1. 创建用来存储您所需的安装资产的安装目录：

```
$ mkdir <installation_directory>
```



重要

您必须创建目录。一些安装信息，如 bootstrap X.509 证书，有较短的过期间隔，因此不要重复使用安装目录。如果要重复使用另一个集群安装中的个别文件，可以将其复制到您的目录中。但是，一些安装数据的文件名可能会在发行版本之间有所改变。从 OpenShift Container Platform 老版本中复制安装文件时要格外小心。

2. 自定义以下 **install-config.yaml** 文件模板，并将它保存到 **<installation_directory>** 中。

注意

此配置文件必须命名为 **install-config.yaml**。

- 除非使用 RHCOS 默认信任的 registry，如 **docker.io**，否则必须在 **additionalTrustBundle** 部分中提供镜像存储库的证书内容。在大多数情况下，必须为您的镜像提供证书。
- 您必须包含命令输出中的 **imageContentSources** 部分，才能镜像存储库。

3. 备份 **install-config.yaml** 文件，以便用于安装多个集群。

重要

install-config.yaml 文件会在安装过程的下一步骤中消耗掉。现在必须备份它。

7.2.6.1. 安装配置参数

在部署 OpenShift Container Platform 集群前，您可以提供参数值，以描述托管集群的云平台的帐户并选择性地自定义集群平台。在创建 **install-config.yaml** 安装配置文件时，您可以通过命令行来提供所需的参数的值。如果要自定义集群，可以修改 **install-config.yaml** 文件来提供关于平台的更多信息。

注意

安装之后，您无法修改 **install-config.yaml** 文件中的这些参数。

重要

openshift-install 命令不验证参数的字段名称。如果指定了不正确的名称，则不会创建相关的文件或对象，且不会报告错误。确保所有指定的参数的字段名称都正确。

7.2.6.1.1. 所需的配置参数

下表描述了所需的安装配置参数：

表 7.18. 所需的参数

参数	描述	值
apiVersion	**install-config.yaml** 内容的 API 版本。当前版本是 **v1**。安装程序还可能支持旧的 API 版本。	字符串
baseDomain	云供应商的基域。此基础域用于创建到 OpenShift Container Platform 集群组件的路由。集群的完整 DNS 名称是 **baseDomain** 和 **metadata.name** 参数值的组合，其格式为 **<metadata.name>.<baseDomain>**。	完全限定域名或子域名，如 **example.com**。
metadata	Kubernetes 资源 **ObjectMeta**，其中只消耗 **name** 参数。	对象

参数	描述	值
metadata.name	集群的名称。集群的 DNS 记录是 **{{.metadata.name}}.{{.baseDomain}}** 的子域。	小写字母,连字符(-)和句点(.)的字符串，如 **dev**。
platform	执行安装的具体平台配置：**aws**、**baremetal**、**azure**、**openstack**、**ovirt**、**vsphere**。有关 **platform.<platform>** 参数的额外信息，请参考下表来了解您的具体平台。	对象

pullSecret	从 Red Hat OpenShift Cluster Manager 获取 pull secret，验证从 Quay.io 等服务中下载 OpenShift Container Platform 组件的容器镜像。	<pre>{
		  "auths":{
		  "cloud.openshift.com":{
		  "auth":"b3Blb=",
		  "email":"you@example.com"
		  },
		  "quay.io":{
		  "auth":"b3Blb=",
		  "email":"you@example.com"
		  }
		  }
		}</pre>

7.2.6.1.2. 网络配置参数

您可以根据现有网络基础架构的要求自定义安装配置。例如，您可以扩展集群网络的 IP 地址块，或者提供不同于默认值的不同 IP 地址块。

只支持 IPv4 地址。

表 7.19. 网络参数

参数	描述	值
networking	集群网络的配置。	对象 **注意** 您不能在安装后修改 **networking** 对象指定的参数。
networking.networkType	要安装的集群网络供应商 Container Network Interface（CNI）插件。	**OpenShiftSDN** 或 **OVNKubernetes**。默认值为 **OpenShiftSDN**。

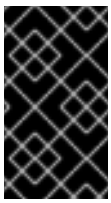
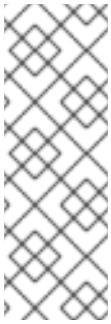
参数	描述	值
networking.clusterNetwork	pod 的 IP 地址块。 默认值为 **10.128.0.0/14**，主机前缀为 **/23**。 如果您指定多个 IP 地址块，则块不得互相重叠。	一个对象数组。例如： networking: clusterNetwork: - cidr: 10.128.0.0/14 hostPrefix: 23
networking.clusterNetwork.cidr	使用 **networking.clusterNetwork** 时需要此项。IP 地址块。 一个 IPv4 网络。	使用 CIDR 形式的 IP 地址块。IPv4 块的前缀长度介于 **0** 到 **32** 之间。
networking.clusterNetwork.hostPrefix	分配给每个单独节点的子网前缀长度。例如，如果 **hostPrefix** 设为 **23**，则每个节点从所给的 **cidr** 中分配一个 **/23** 子网。**hostPrefix** 值 **23** 提供 510（2^(32 - 23)- 2）个 pod IP 地址。	子网前缀。 默认值为 **23**。
networking.serviceNetwork	服务的 IP 地址块。默认值为 **172.30.0.0/16**。 OpenShift SDN 和 OVN-Kubernetes 网络供应商只支持服务网络的一个 IP 地址块。	CIDR 格式具有 IP 地址块的数组。例如： networking: serviceNetwork: - 172.30.0.0/16
networking.machineNetwork	机器的 IP 地址块。 如果您指定多个 IP 地址块，则块不得互相重叠。 如果您指定了多个 IP 内核参数，**machineNetwork.cidr** 值必须是主网络的 CIDR。	一个对象数组。例如： networking: machineNetwork: - cidr: 10.0.0.0/16
networking.machineNetwork.cidr	使用 **networking.machineNetwork** 时需要。IP 地址块。libvirt 以外的所有平台的默认值为 **10.0.0.0/16**。对于 libvirt，默认值为 **192.168.126.0/24**。	CIDR 表示法中的 IP 网络块。 例如：**10.0.0.0/16**。  **注意** 将 **networking.machineNetwork** 设置为与首选 NIC 所在的 CIDR 匹配。

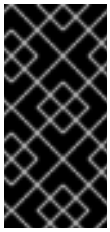
7.2.6.1.3. 可选配置参数

下表描述了可选安装配置参数：

表 7.20. 可选参数

参数	描述	值
additionalTrustBundle	添加到节点可信证书存储中的 PEM 编码 X.509 证书捆绑包。配置了代理时，也可以使用这个信任捆绑包。	字符串
compute	组成计算节点的机器的配置。	machine-pool 对象的数组。详情请查看以下"Machine-pool"表。
compute.architecture	决定池中机器的指令集合架构。目前不支持异构集群，因此所有池都必须指定相同的架构。有效值为 **amd64** （默认值）。	字符串
compute.hyperthreading	是否在计算机器上启用或禁用并发多线程或**超线程**。默认情况下，启用并发多线程以提高机器内核的性能。  **重要** 如果禁用并发多线程，请确保在容量规划时考虑到机器性能可能会显著降低的问题。	**Enabled** 或 **Disabled**
compute.name	使用 **compute** 时需要此值。机器池的名称。	**worker**
compute.platform	使用 **compute** 时需要此值。使用此参数指定托管 worker 机器的云供应商。此参数值必须与 **controlPlane.platform** 参数值匹配。	**aws**、**azure**、**gcp**、**openstack**、**ovirt**、**vsphere** 或 **{}**
compute.replicas	要置备的计算机器数量，也称为 worker 机器。	大于或等于 **2** 的正整数。默认值为 **3**。
controlPlane	组成 control plane 的机器的配置。	**MachinePool** 对象的数组。详情请查看以下"Machine-pool"表。
controlPlane.architecture	决定池中机器的指令集合架构。目前不支持异构集群，因此所有池都必须指定相同的架构。有效值为 **amd64** （默认值）。	字符串

参数	描述	值
controlPlane.hyperthreading	是否在 control plane 机器上启用或禁用并发多线程或**超线程**。默认情况下，启用并发多线程以提高机器内核的性能。  **重要** 如果禁用并发多线程，请确保在容量规划时考虑到机器性能可能会显著降低的问题。	**Enabled** 或 **Disabled**
controlPlane.name	使用 **controlPlane** 时需要。机器池的名称。	**master**
controlPlane.platform	使用 **controlPlane** 时需要。使用此参数指定托管 control plane 机器的云供应商。此参数值必须与 **compute.platform** 参数值匹配。	**aws**、**azure**、**gcp**、**openstack**、**ovirt**、**vsphere** 或 **{}**
controlPlane.replicas	要置备的 control plane 机器数量。	唯一支持的值是 **3**，它是默认值。
credentialsMode	Cloud Credential Operator（CCO）模式。如果没有指定任何模式，CCO 会动态地尝试决定提供的凭证的功能，在支持多个模式的平台上使用 mint 模式。  **注意** 不是所有 CCO 模式都支持所有云供应商。如需有关 CCO 模式的更多信息，请参阅*Red Hat Operator 参考指南*内容中的 *Cloud Credential Operator* 条目。	**Mint**、**Passthrough**、**Manual** 或空字符串(**""**)。

参数	描述	值
fips	启用或禁用 FIPS 模式。默认为 **false**（禁用）。如果启用了 FIPS 模式，运行 OpenShift Container Platform 的 Red Hat Enterprise Linux CoreOS (RHCOS) 机器会绕过默认的 Kubernetes 加密套件，并使用由 RHCOS 提供的加密模块。  **重要** 只有在 **x86_64** 架构中的 OpenShift Container Platform 部署支持 FIPS 验证的/Modules in Process 加密库。  **注意** 如果使用 Azure File 存储，则无法启用 FIPS 模式。	**false** 或 **true**
imageContentSources	release-image 内容的源和仓库。	对象数组。包括一个 **source** 以及可选的 **mirrors**，如下表所示。
imageContentSources.source	使用 **imageContentSources** 时需要。指定用户在镜像拉取规格中引用的仓库。	字符串
imageContentSources.mirrors	指定可能还包含同一镜像的一个或多个仓库。	字符串数组
publish	如何发布或公开集群的面向用户的端点，如 Kubernetes API、OpenShift 路由。	**Internal** 或 **External**。默认值为 **External**。 在非云平台上不支持将此字段设置为 **Internal**。  **重要** 如果将字段的值设为 **Internal**，集群将无法运行。如需更多信息，请参阅 BZ#1953035。

参数	描述	值
sshKey	用于验证集群机器访问的 SSH 密钥或密钥。 **注意** 对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 **ssh-agent** 进程使用的 SSH 密钥。	一个或多个密钥。例如： sshKey: <key1> <key2> <key3>

7.2.6.2. IBM Z 的 install-config.yaml 文件示例

您可以自定义 **install-config.yaml** 文件，以指定有关 OpenShift Container Platform 集群平台的更多信息，或修改所需参数的值。

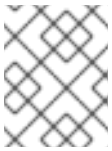
```
apiVersion: v1
baseDomain: example.com (1)
compute: (2)
- hyperthreading: Enabled (3)
  name: worker
  replicas: 0 (4)
  architecture : s390x
controlPlane: (5)
  hyperthreading: Enabled (6)
  name: master
  replicas: 3 (7)
  architecture : s390x
metadata:
  name: test (8)
networking:
  clusterNetwork:
  - cidr: 10.128.0.0/14 (9)
    hostPrefix: 23 (10)
  networkType: OpenShiftSDN
  serviceNetwork: (11)
  - 172.30.0.0/16
platform:
  none: {} (12)
fips: false (13)
pullSecret: '{"auths":{"<local_registry>": {"auth": "<credentials>","email": "you@example.com"}}}' (14)
sshKey: 'ssh-ed25519 AAAA...' (15)
additionalTrustBundle: | (16)
  -----BEGIN CERTIFICATE-----
  ZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZ
  -----END CERTIFICATE-----
imageContentSources: (17)
```

```
- mirrors:
  - <local_repository>/ocp4/openshift4
  source: quay.io/openshift-release-dev/ocp-release
- mirrors:
  - <local_repository>/ocp4/openshift4
  source: quay.io/openshift-release-dev/ocp-v4.0-art-dev
```

1 集群的基域。所有 DNS 记录都必须是这个基域的子域，并包含集群名称。

2 5 **controlPlane** 部分是一个单映射，但 **compute** 部分是一系列映射。为满足不同数据结构的要求，**compute** 部分的第一行必须以连字符 **-** 开头，**controlPlane** 部分的第一行则不可以连字符开头。只使用一个 control plane 池。

3 6 是否要启用或禁用并发多线程（SMT）或**超线程**。默认情况下，启用 SMT 可提高机器内核的性能。您可以通过将参数值设为 **Disabled** 来禁用。如果禁用 SMT，则必须在所有集群机器中禁用它，其中包括 control plane 和计算机器。



注意

默认启用并发多线程（SMT）。如果在 BIOS 设置中没有启用 SMT，**hyperthreading** 参数不会起作用。

重要

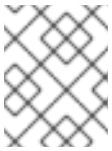
如果您禁用 **hyperthreading**（无论是在 BIOS 中还是在 **install-config.yaml** 中），请确保您对可能会造成的机器性能显著降低的情况有所考虑。

4 **replicas** 参数的值必须设置为 **0**。此参数控制集群为您创建和管理的 worker 数量，使用用户置备的基础架构时集群不会执行这些功能。在完成 OpenShift Container Platform 安装前，您必须手动为集群部署 worker 机器。

7 您添加到集群的 control plane 机器数量。由于集群将这个值用作集群中 etcd 端点的数量，因此该值必须与您部署的 control plane 机器数量匹配。

8 您在 DNS 记录中指定的集群名称。

9 从中分配 pod IP 地址的 IP 地址块。此块不得与现有的物理网络重叠。这些 IP 地址用于 pod 网络。如果您需要从外部网络访问 pod，请配置负载均衡器和路由器来管理流量。



注意

类 E CIDR 范围保留给以后使用。要使用 Class E CIDR 范围，您必须确保您的网络环境接受 Class E CIDR 范围内的 IP 地址。

10 分配给每个单独节点的子网前缀长度。例如，如果 **hostPrefix** 设为 **23**，则每个节点从所给的 **cidr** 中分配一个 **/23** 子网，这样就能有 510 (2^(32 - 23) - 2) 个 Pod IP 地址。如果您需要从外部网络访问节点，请配置负载均衡器和路由器来管理流量。

11 用于服务 IP 地址的 IP 地址池。您只能输入一个 IP 地址池。此块不得与现有的物理网络重叠。如果您需要从外部网络访问服务，请配置负载均衡器和路由器来管理流量。

12 您必须将平台设置为 **none**。您无法为 IBM Z 基础架构提供额外的平台配置变量。

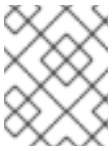
13 是否启用或禁用 FIPS 模式。默认情况下不启用 FIPS 模式。如果启用了 FIPS 模式，运行 OpenShift

重要

只有在 **x86_64** 架构中的 OpenShift Container Platform 部署支持 FIPS 验证的/Modules in Process 加密库。

14 对于 **<local_registry>**，请指定 registry 域名，以及您的镜像 registry 用来提供内容的可选端口。例如： **registry.example.com** 或者 **registry.example.com:5000**。使用 **<credentials>** 为您生成的镜像 registry 指定 base64 编码的用户名和密码。

15 Red Hat Enterprise Linux CoreOS (RHCOS) 中 **core** 用户的默认 SSH 密钥的公钥部分。



注意

对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 **ssh-agent** 进程使用的 SSH 密钥。

16 添加 **additionalTrustBundle** 参数和值。该值必须是您用于镜像 registry 的证书文件内容，可以是现有的可信证书颁发机构或您为镜像 registry 生成的自签名证书。

17 提供命令输出中的 **imageContentSources** 部分来镜像存储库。

7.2.6.3. 在安装过程中配置集群范围代理

生产环境可能会拒绝直接访问互联网，而是提供 HTTP 或 HTTPS 代理。您可以通过在 **install-config.yaml** 文件中配置代理设置，将新的 OpenShift Container Platform 集群配置为使用代理。

先决条件

- 您有一个现有的 **install-config.yaml** 文件。
- 您检查了集群需要访问的站点，并决定是否需要绕过代理。默认情况下代理所有集群出口流量，包括对托管云供应商 API 的调用。您需要将站点添加到 **Proxy** 对象的 **spec.noProxy** 字段来绕过代理。

注意

Proxy 对象 **status.noProxy** 字段使用安装配置中的 **networking.machineNetwork[].cidr**、**networking.clusterNetwork[].cidr** 和 **networking.serviceNetwork[]** 字段的值填充。

对于在 Amazon Web Services(AWS)、Google Cloud Platform(GCP)、Microsoft Azure 和 Red Hat OpenStack Platform(RHOSP)上安装, **Proxy** 对象 **status.noProxy** 字段也会使用实例元数据端点填充(**169.254.169.254**)。

流程

1. 编辑 **install-config.yaml** 文件并添加代理设置。例如：

```
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
```

```
  noProxy: example.com 3
additionalTrustBundle: | 4
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
...
```

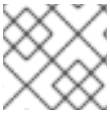
1. 用于创建集群外 HTTP 连接的代理 URL。URL 必须是 **http**。
2. 用于创建集群外 HTTPS 连接的代理 URL。
3. 要排除在代理中的目标域名、IP 地址或其他网络 CIDR 的逗号分隔列表。在域前面加 **.** 来仅匹配子域。例如： **.y.com** 匹配 **x.y.com**，但不匹配 **y.com**。使用 * 绕过所有目的地的代理。
4. 如果提供，安装程序会在 **openshift-config 命名空间中生成名为 user-ca- bundle** 的配置映射来保存额外的 CA 证书。如果您提供 **additionalTrustBundle** 和至少一个代理设置，则 **Proxy** 对象会被配置为引用 **trustedCA** 字段中的 **user-ca-bundle** 配置映射。然后，Cluster Network Operator 会创建一个 **trusted-ca-bundle** 配置映射，该配置映射将为 **trustedCA** 参数指定的内容与 RHCOS 信任捆绑包合并。 **additionalTrustBundle** 字段是必需的，除非代理的身份证书由来自 RHCOS 信任捆绑包的颁发机构签名。

注意

安装程序不支持代理的 **readinessEndpoints** 字段。

2. 保存该文件，并在安装 OpenShift Container Platform 时引用。

安装程序会创建一个名为 **cluster** 的集群范围代理，该代理使用提供的 **install-config.yaml** 文件中的代理设置。如果没有提供代理设置，仍然会创建一个 **cluster Proxy** 对象，但它会有一个空 **spec**。



注意

只支持名为 **cluster** 的 **Proxy** 对象，且无法创建额外的代理。

7.2.7. 创建 Kubernetes 清单和 Ignition 配置文件

由于您必须修改一些集群定义文件并要手动启动集群机器，因此您必须生成 Kubernetes 清单和 Ignition 配置文件，集群需要这两项来创建其机器。

安装配置文件转换为 Kubernetes 清单。清单嵌套到 Ignition 配置文件中，稍后用于创建集群。

重要

- 安装程序生成的 Ignition 配置文件包含在 24 小时后过期的证书，然后在过期时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外情况是，您需要手动批准待处理的 **node-bootstrapper** 证书签名请求（CSR）来恢复 kubelet 证书。如需更多信息，请参阅 *从过期的 control plane 证书中恢复*的文档。
- 建议您在生成 12 小时后使用 Ignition 配置文件，因为集群安装后 24 小时证书从 16 小时轮转至 22 小时。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中运行证书更新时避免安装失败。

先决条件

- 已获得 OpenShift Container Platform 安装程序。
- 已创建 **install-config.yaml** 安装配置文件。

流程

1. 切换到包含安装程序的目录，并为集群生成 Kubernetes 清单：

```
$ ./openshift-install create manifests --dir <installation_directory> 1
```

1 对于 **<installation_directory>**，请指定含有您创建的 **install-config.yaml** 文件的安装目录。

2. 检查 **<installation_directory>/manifests/cluster-scheduler-02-config.yml** Kubernetes 清单文件中的 **mastersSchedulable** 参数是否已设置为 **false**。此设置可防止在 control plane 机器上调度 pod:
   a. 打开 **<installation_directory>/manifests/cluster-scheduler-02-config.yml** 文件。
   b. 找到 **mastersSchedulable** 参数并确保它被设置为 **false**。
   c. 保存并退出文件。
3. 要创建 Ignition 配置文件，从包含安装程序的目录运行以下命令：

```
$ ./openshift-install create ignition-configs --dir <installation_directory> 1
```

1 对于 **<installation_directory>**，请指定相同的安装目录。

该目录中将生成以下文件：

```
.
├── auth
│   ├── kubeadmin-password
│   └── kubeconfig
├── bootstrap.ign
├── master.ign
├── metadata.json
└── worker.ign
```

7.2.8. 创建 Red Hat Enterprise Linux CoreOS (RHCOS) 机器

在您置备的 IBM Z 环境中安装集群前，您必须在 z/VM 虚拟机上安装 RHCOS 以便集群使用。完成以下步骤以创建机器。

先决条件

- 在置备机器中运行 FTP 服务器，您创建的机器需要可以访问这个 FTP 服务器。

流程

1. 在您置备的机器上登录到 Linux。
2. 从 RHCOS 镜像镜像 获取 Red Hat Enterprise Linux CoreOS（RHCOS）内核、initramfs 和 rootfs 文件。

重要

RHCOS 镜像可能不会随着 OpenShift Container Platform 的每一发行版本都有改变。您必须下载最高版本的镜像，其版本号应小于或等于您安装的 OpenShift Container Platform 版本。只使用以下流程中描述的适当内核、initramfs 和 rootfs 工件。

文件名包含 OpenShift Container Platform 版本号。它们类似以下示例：

- kernel: **rhcos-<version>-live-kernel-<architecture>**
- initramfs: **rhcos-<version>-live-initramfs.<architecture>.img**
- rootfs: **rhcos-<version>-live-rootfs.<architecture>.img**

注意

FCP 和 DASD 的 rootfs 镜像是相同的。

3. 创建参数文件。以下参数特定于特定虚拟机：
   - 对于 **coreos.inst.install_dev=**，请为 DASD 安装指定 **dasda**，或者为 FCP 指定 **sda**。请注意 FCP 需要 **zfcp.allow_lun_scan=0**。
   - 对于 **rd.dasd=**，请指定要安装 RHCOS 的 DASD。
   - **rd.zfcp=<adapter>,<wwpn>,<lun>** 指定要在其中安装 RHCOS 的 FCP 磁盘。
   - 对于 **ip=**，请指定以下七项：
     i. 机器的 IP 地址。
     ii. 一个空字符串。
     iii. 网关
     iv. 子网掩码。
     v. **hostname.domainname**格式的机器主机和域名。省略这个值会让 RHCOS 来决定这个值。
     vi. 网络接口名称。省略这个值会让 RHCOS 来决定这个值。
     vii. 如果使用静态 IP 地址，则为一个空字符串。
   - 对于 **coreos.inst.ignition_url=**，为机器角色指定 Ignition 文件。使用 **bootstrap.ign**、**master.ign** 或 **worker.ign**。只支持 HTTP 和 HTTPS 协议。
   - 对于 **coreos.live.rootfs_url=**，为您引导的内核和 initramfs 指定匹配的 rootfs 工件。只支持 HTTP 和 HTTPS 协议。

- 所有其他参数都可以保留。
  bootstrap 机器的实例参数文件（**bootstrap-0.parm**）如下：

```
rd.neednet=1 \
console=ttysclp0 \
coreos.inst.install_dev=dasda \
coreos.live.rootfs_url=http://cl1.provide.example.com:8080/assets/rhcos-live-
rootfs.s390x.img \
coreos.inst.ignition_url=http://cl1.provide.example.com:8080/ignition/bootstrap.ign \
ip=172.18.78.2::172.18.78.1:255.255.255.0:::none nameserver=172.18.78.1 \
rd.znet=qeth,0.0.bdf0,0.0.bdf1,0.0.bdf2,layer2=1,portno=0 \
zfcp.allow_lun_scan=0 \
rd.dasd=0.0.3490
```

  将参数文件中的所有选项写为一行，并确保您没有换行字符。

4. 将 initramfs 、内核、参数文件和 RHCOS 镜像传送到 z/VM 中，例如使用 FTP。有关如何使用 FTP 传输文件并从虚拟 reader 引导的详情，请参考 在 Z/VM 中安装。

5. 将文件 punch 到 z/VM 虚拟机的虚拟 reader，即成为 bootstrap 节点。
   请参阅 IBM 文档中的 PUNCH。

   提示

   您可以使用 CP PUNCH 命令（如果是 Linux，使用 **vmur** 命令）在两个 z/VM 虚拟机间传输文件。

6. 在 bootstrap 机器中登录到 CMS。

7. 从 reader IPL bootstrap 机器：

```
$ ipl c
```

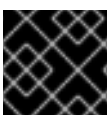
   请参阅 IBM 文档中的 IPL。

8. 对集群中的其他机器重复此步骤。

7.2.8.1. 高级 RHCOS 安装参考

本节演示了网络配置和其他高级选项，允许您修改 Red Hat Enterprise Linux CoreOS（RHCOS）手动安装过程。下表描述了您可以与 RHCOS live installer 和 **coreos-installer** 命令一起使用的内核参数和命令行选项。

RHCOS 启动提示下的路由和绑定选项
如果从 ISO 镜像安装 RHCOS，您可以在引导该镜像时手动添加内核参数以配置节点的网络。如果没有使用网络参数，则安装默认为使用 DHCP。



重要

添加网络参数时，还必须添加 **rd.neednet=1** 内核参数。

下表描述了如何为实时 ISO 安装使用 **ip=** 、 **nameserver=** 和 **bond=** 内核参数。

注意

在添加内核参数时顺序非常重要： **ip=**，**nameserver=**，然后 **bond=**。

ISO 的路由和绑定选项

下表提供了配置 Red Hat Enterprise Linux CoreOS（RHCOS）节点网络的示例。这些是在系统引导过程中传递给 **dracut** 工具的网络选项。有关 **dracut** 支持的网络选项的详情，请参考 **dracut.cmdline** 手册页。

描述	例子
要配置一个 IP 地址，可以使用 DHCP(**ip=dhcp**)或者设置单独的静态 IP 地址(**ip=<host_ip>**)。然后在每个节点上指定 DNS 服务器 IP 地址(**nameserver=<dns_ip>**)。这个示例设置： • 节点的 IP 地址为 **10.10.10.2** • 网关地址为 **10.10.10.254** • 子网掩码为 **255.255.255.0** • 主机名为 **core0.example.com** • DNS 服务器地址为 **4.4.4.41**	ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp1s0:none nameserver=4.4.4.41
通过指定多个 **ip=** 条目来指定多个网络接口。	ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp1s0:none ip=10.10.10.3::10.10.10.254:255.255.255.0:core0.example.com:enp2s0:none
可选：您可以通过设置一个 **rd.route=** 值来配置到额外网络的路由。 如果额外网络网关与主要网络网关不同，则默认网关必须是主要网络网关。	配置默认网关： ip=::10.10.10.254:::: 为额外网络配置路由： rd.route=20.20.20.0/24:20.20.20.254:enp2s0
在单一接口中禁用 DHCP，比如当有两个或者多个网络接口时，且只有一个接口被使用。	ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp1s0:none ip=::::core0.example.com:enp2s0:none
您可以将系统中 DHCP 和静态 IP 配置与多个网络接口结合在一起。	ip=enp1s0:dhcp ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp2s0:none

描述	例子
可选： 您可以使用 **vlan=** 参数在单独的接口上配置 VLAN。	在网络接口中配置 VLAN 并使用静态 IP 地址： ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp2s0.100:none vlan=enp2s0.100:enp2s0 在网络接口中配置 VLAN 并使用 DHCP： ip=enp2s0.100:dhcp vlan=enp2s0.100:enp2s0
您可以为每个服务器添加一个 **nameserver=** 条目来提供多个 DNS 服务器。	nameserver=1.1.1.1 nameserver=8.8.8.8
可选：使用 **bond=** 选项支持将多个网络接口绑定到一个接口。在这两个示例中： - 配置绑定接口的语法为：**bond=name[:network_interfaces][:options]** - *name* 是绑定设备名称（**bond0**），*network_interfaces* 代表用逗号分开的物理（以太网）接口(**em1,em2**)的列表，*options* 是用逗号分开的绑定选项列表。输入 **modinfo bonding** 查看可用选项。 - 当使用 **bond=** 创建绑定接口时，您必须指定如何分配 IP 地址以及绑定接口的其他信息。	要将绑定的接口配置为使用 DHCP，请将绑定的 IP 地址设置为 **dhcp**。例如： bond=bond0:em1,em2:mode=active-backup ip=bond0:dhcp 要将绑定接口配置为使用静态 IP 地址，请输入您需要的特定 IP 地址以及相关信息。例如： bond=bond0:em1,em2:mode=active-backup ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:bond0:none
可选： 您可以使用 **vlan=** 参数在绑定接口上配置 VLAN。	使用 VLAN 配置绑定接口并使用 DHCP： ip=bond0.100:dhcp bond=bond0:em1,em2:mode=active-backup vlan=bond0.100:bond0 使用 VLAN 配置绑定接口，并使用静态 IP 地址： ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:bond0.100:none bond=bond0:em1,em2:mode=active-backup vlan=bond0.100:bond0

描述	例子
可选：通过使用 **team=** 参数，网络合作可用作绑定的替代选择。在此例中： • 配置组接口的语法为：**team=name[:network_interfaces]** *name* 是组设备名称(**team0**)，**network** *_interfaces* 代表以逗号分隔的物理（以太网）接口（**em1**、**em2**）列表。 **注意** 当 RHCOS 切换到即将推出的 RHEL 版本时，团队计划会被弃用。如需更多信息，请参阅红帽知识库文章。	配置网络团队： team=team0:em1,em2 ip=team0:dhcp

7.2.9. 创建集群

要创建 OpenShift Container Platform 集群，请等待您通过安装程序生成的 Ignition 配置文件所置备的机器上完成 bootstrap 过程。

先决条件

- 为集群创建所需的基础架构。
- 已获得安装程序并为集群生成了 Ignition 配置文件。
- 已使用 Ignition 配置文件为集群创建 RHCOS 机器。

流程

1. 监控 bootstrap 过程：

```
$ ./openshift-install --dir <installation_directory> wait-for bootstrap-complete \ 1
    --log-level=info 2
```

1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

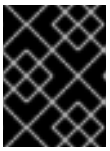
2 要查看不同的安装详情，请指定 **warn**、**debug** 或 **error**，而不要指定 **info**。

输出示例

```
INFO Waiting up to 30m0s for the Kubernetes API at https://api.test.example.com:6443...
INFO API v1.19.0 up
INFO Waiting up to 30m0s for bootstrapping to complete...
INFO It is now safe to remove the bootstrap resources
```

Kubernetes API 服务器提示已在 control plane 机器上完成 bootstrap 时，命令运行成功。

2. bootstrap 过程完成后，请从负载均衡器中删除 bootstrap 机器。



重要

此时您必须从负载均衡器中删除 bootstrap 机器。您还可以删除或重新格式化机器本身。

7.2.10. 使用 CLI 登录到集群

您可以通过导出集群 **kubeconfig** 文件，以默认系统用户身份登录集群。**kubeconfig** 文件包含关于集群的信息，供 CLI 用于将客户端连接到正确集群和 API 服务器。该文件特只适用于一个特定的集群，在 OpenShift Container Platform 安装过程中创建。

先决条件

- 已部署了 OpenShift Container Platform 集群。
- 已安装 **oc** CLI。

流程

1. 导出 **kubeadmin** 凭证：

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

2. 使用导出的配置，验证能否成功运行 **oc** 命令：

```
$ oc whoami
```

输出示例

```
system:admin
```

7.2.11. 批准机器的证书签名请求

将机器添加到集群时，会为您添加的每台机器生成两个待处理证书签名请求（CSR）。您必须确认这些 CSR 已获得批准，或根据需要自行批准。客户端请求必须首先被批准，然后是服务器请求。

先决条件

- 您已将机器添加到集群中。

流程

1. 确认集群可以识别这些机器：

```
$ oc get nodes
```

输出示例

```
NAME      STATUS    ROLES   AGE  VERSION
```

```
master-0  Ready    master  63m  v1.19.0
master-1  Ready    master  63m  v1.19.0
master-2  Ready    master  64m  v1.19.0
```

输出将列出您创建的所有机器。

注意

在一些 CSR 被批准前，以上输出可能不包括计算节点（也称为 worker 节点）。

2. 检查待处理的 CSR，并确保可以看到添加到集群中的每台机器都有 **Pending** 或 **Approved** 状态的客户端请求：

```
$ oc get csr
```

输出示例

```
NAME        AGE    REQUESTOR                                                   CONDITION
csr-8b2br   15m    system:serviceaccount:openshift-machine-config-operator:node-
bootstrapper   Pending
csr-8vnps   15m    system:serviceaccount:openshift-machine-config-operator:node-
bootstrapper   Pending
...
```

在本例中，两台机器加入了集群。您可能会在列表中看到更多已批准的 CSR。

3. 如果 CSR 没有获得批准，请在所添加机器的所有待处理 CSR 都处于 **Pending** 状态后，为您的集群机器批准这些 CSR：

注意

由于 CSR 会自动轮转，因此请在将机器添加到集群后一小时内批准您的 CSR。如果没有在一小时内批准，证书将会轮转，每个节点将会存在多个证书。您必须批准所有这些证书。批准客户端 CSR 后， Kubelet 为服务证书创建辅助 CSR，这需要手动批准。然后，如果 Kubelet 请求具有相同参数的新证书，则 **machine-approver** 会自动批准后续服务证书续订请求。

注意

对于在未启用机器 API 的平台中运行的集群，如裸机和其他用户置备的基础架构，必须采用一种方法自动批准 kubelet 提供证书请求（CSR）。如果没有批准请求，则 **oc exec**、**oc rsh** 和 **oc logs** 命令将无法成功，因为 API 服务器连接到 kubelet 时需要服务证书。与 Kubelet 端点联系的任何操作都需要此证书批准。这个方法必须监视新的 CSR，确认 CSR 由 **system:node** 或 **system:admin** 组中的 **node-bootstrapper** 服务帐户提交，并确认节点的身份。

- 若要单独批准，请对每个有效的 CSR 运行以下命令：

```
$ oc adm certificate approve <csr_name> 1
```

1 **<csr_name>** 是当前 CSR 列表中 CSR 的名称。

- 要批准所有待处理的 CSR，请运行以下命令：

```
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}{{"\n"}}
{{end}}{{end}}' | xargs --no-run-if-empty oc adm certificate approve
```

注意

在有些 CSR 被批准前，一些 Operator 可能无法使用。

4. 现在，您的客户端请求已被批准，您必须查看添加到集群中的每台机器的服务器请求：

```
$ oc get csr
```

输出示例

```
NAME        AGE     REQUESTOR                                                   CONDITION
csr-bfd72   5m26s   system:node:ip-10-0-50-126.us-east-2.compute.internal
Pending
csr-c57lv   5m26s   system:node:ip-10-0-95-157.us-east-2.compute.internal
Pending
...
```

5. 如果剩余的 CSR 没有被批准，且处于 **Pending** 状态，请批准集群机器的 CSR：

- 若要单独批准，请对每个有效的 CSR 运行以下命令：

```
$ oc adm certificate approve <csr_name> 1
```

1 **<csr_name>** 是当前 CSR 列表中 CSR 的名称。

- 要批准所有待处理的 CSR，请运行以下命令：

```
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}{{"\n"}}
{{end}}{{end}}' | xargs oc adm certificate approve
```

6. 批准所有客户端和服务器 CSR 后，器将处于 **Ready** 状态。运行以下命令验证：

```
$ oc get nodes
```

输出示例

```
NAME      STATUS    ROLES   AGE  VERSION
master-0  Ready     master  73m  v1.20.0
master-1  Ready     master  73m  v1.20.0
master-2  Ready     master  74m  v1.20.0
worker-0  Ready     worker  11m  v1.20.0
worker-1  Ready     worker  11m  v1.20.0
```

注意

批准服务器 CSR 后可能需要几分钟时间让机器转换为 **Ready** 状态。

其他信息

- 如需有关 CSR 的更多信息，请参阅证书签名请求。

7.2.12. 初始 Operator 配置

在 control plane 初始化后，您必须立即配置一些 Operator 以便它们都可用。

先决条件

- 您的 control plane 已初始化。

流程

1. 观察集群组件上线：

```
$ watch -n5 oc get clusteroperators
```

输出示例

```
NAME                                       VERSION AVAILABLE   PROGRESSING   DEGRADED
SINCE
authentication                             4.6.0   True        False         False      3h56m
cloud-credential                           4.6.0   True        False         False      29h
cluster-autoscaler                         4.6.0   True        False         False      29h
config-operator                            4.6.0   True        False         False      6h39m
console                                    4.6.0   True        False         False      3h59m
csi-snapshot-controller                    4.6.0   True        False         False      4h12m
dns                                        4.6.0   True        False         False      4h15m
etcd                                       4.6.0   True        False         False      29h
image-registry                             4.6.0   True        False         False      3h59m
ingress                                    4.6.0   True        False         False      4h30m
insights                                   4.6.0   True        False         False      29h
kube-apiserver                             4.6.0   True        False         False      29h
kube-controller-manager                    4.6.0   True        False         False      29h
kube-scheduler                             4.6.0   True        False         False      29h
kube-storage-version-migrator              4.6.0   True        False         False      4h2m
machine-api                                4.6.0   True        False         False      29h
machine-approver                           4.6.0   True        False         False      6h34m
machine-config                             4.6.0   True        False         False      3h56m
marketplace                                4.6.0   True        False         False      4h2m
monitoring                                 4.6.0   True        False         False      6h31m
network                                    4.6.0   True        False         False      29h
node-tuning                                4.6.0   True        False         False      4h30m
openshift-apiserver                        4.6.0   True        False         False      3h56m
openshift-controller-manager               4.6.0   True        False         False      4h36m
openshift-samples                          4.6.0   True        False         False      4h30m
operator-lifecycle-manager                 4.6.0   True        False         False      29h
operator-lifecycle-manager-catalog         4.6.0   True        False         False      29h
operator-lifecycle-manager-packageserver   4.6.0   True        False         False      3h59m
service-ca                                 4.6.0   True        False         False      29h
storage                                    4.6.0   True        False         False      4h30m
```

2. 配置不可用的 Operator。

7.2.12.1. 禁用默认的 OperatorHub 源

在 OpenShift Container Platform 安装过程中，默认为 OperatorHub 配置由红帽和社区项目提供的源内容的 operator 目录。在受限网络环境中，必须以集群管理员身份禁用默认目录。

流程

- 通过在 **OperatorHub** 对象中添加 **disableAllDefaultSources: true** 来禁用默认目录的源：

```
$ oc patch OperatorHub cluster --type json \
    -p '[{"op": "add", "path": "/spec/disableAllDefaultSources", "value": true}]'
```

提示

或者，您可以使用 Web 控制台管理目录源。在 **Administration → Cluster Settings → Global Configuration → OperatorHub** 页面中，点 **Sources** 选项卡，其中可创建、删除、禁用和启用单独的源。

7.2.12.2. 镜像 registry 存储配置

对于不提供默认存储的平台，Image Registry Operator 最初将不可用。安装后，您必须配置 registry 使用的存储，这样 Registry Operator 才可用。

示配置生产集群所需的持久性卷的说明。如果适用，显示有关将空目录配置为存储位置的说明，该位置只可用于非生产集群。

另外还提供了在升级过程中使用 **Recreate** rollout 策略来允许镜像 registry 使用块存储类型的说明。

7.2.12.2.1. 为 IBM Z 配置 registry 存储

作为集群管理员，在安装后需要配置 registry 来使用存储。

先决条件

- 具有 Cluster Administrator 权限
- IBM Z 上的集群。
- 为集群置备的持久性存储。

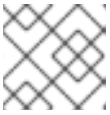
重要

如果您只有一个副本，OpenShift Container Platform 支持对镜像 registry 存储的 **ReadWriteOnce** 访问。要部署支持高可用性的、带有两个或多个副本的镜像 registry，需要 **ReadWriteMany** 访问设置。

- 必须具有 100Gi 容量。

流程

1. 为了配置 registry 使用存储，需要修改 **configs.imageregistry/cluster** 资源中的 **spec.storage.pvc** 。

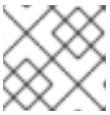


注意

使用共享存储时，请查看您的安全设置以防止被外部访问。

2. 验证您没有 registry pod:

```
$ oc get pod -n openshift-image-registry
```



注意

如果存储类型为 **emptyDIR**，则副本数不能超过 **1**。

3. 检查 registry 配置：

```
$ oc edit configs.imageregistry.operator.openshift.io
```

输出示例

```
storage:
  pvc:
    claim:
```

将 **claim** 字段留空以允许自动创建一个 **image-registry-storage** PVC。

4. 检查 **clusteroperator** 的状态：

```
$ oc get clusteroperator image-registry
```

5. 确保您的 registry 设置为 manage，以启用镜像的构建和推送。

- 运行：

```
$ oc edit configs.imageregistry/cluster
```

然后将行改

```
managementState: Removed
```

为

```
managementState: Managed
```

7.2.12.2.2. 在非生产集群中配置镜像 registry 存储

您必须为 Image Registry Operator 配置存储。对于非生产集群，您可以将镜像 registry 设置为空目录。如果您这样做，重启 registry 后会丢失所有镜像。

流程

- 将镜像 registry 存储设置为空目录：

```
$ oc patch configs.imageregistry.operator.openshift.io cluster --type merge --patch '{"spec":
{"storage":{"emptyDir":{}}}}'
```

警告

仅可为非生产集群配置这个选项。

如果在 Image Registry Operator 初始化其组件前运行此命令，**oc patch** 命令会失败并显示以下错误：

```
Error from server (NotFound): configs.imageregistry.operator.openshift.io "cluster" not found
```

等待几分钟，然后再次运行该命令。

7.2.13. 在用户置备的基础架构上完成安装

完成 Operator 配置后，可以在您提供的基础架构上完成集群安装。

先决条件

- 您的 control plane 已初始化。
- 已完成初始 Operator 配置。

流程

1. 使用以下命令确认所有集群组件都已在线：

```
$ watch -n5 oc get clusteroperators
```

输出示例

```
NAME                                       VERSION AVAILABLE   PROGRESSING   DEGRADED
SINCE
authentication                             4.6.0   True        False         False      3h56m
cloud-credential                           4.6.0   True        False         False      29h
cluster-autoscaler                         4.6.0   True        False         False      29h
config-operator                            4.6.0   True        False         False      6h39m
console                                    4.6.0   True        False         False      3h59m
csi-snapshot-controller                    4.6.0   True        False         False      4h12m
dns                                        4.6.0   True        False         False      4h15m
etcd                                       4.6.0   True        False         False      29h
image-registry                             4.6.0   True        False         False      3h59m
ingress                                    4.6.0   True        False         False      4h30m
insights                                   4.6.0   True        False         False      29h
kube-apiserver                             4.6.0   True        False         False      29h
kube-controller-manager                    4.6.0   True        False         False      29h
kube-scheduler                             4.6.0   True        False         False      29h
```

```
kube-storage-version-migrator              4.6.0   True        False         False      4h2m
machine-api                                4.6.0   True        False         False      29h
machine-approver                           4.6.0   True        False         False      6h34m
machine-config                             4.6.0   True        False         False      3h56m
marketplace                                4.6.0   True        False         False      4h2m
monitoring                                 4.6.0   True        False         False      6h31m
network                                    4.6.0   True        False         False      29h
node-tuning                                4.6.0   True        False         False      4h30m
openshift-apiserver                        4.6.0   True        False         False      3h56m
openshift-controller-manager               4.6.0   True        False         False      4h36m
openshift-samples                          4.6.0   True        False         False      4h30m
operator-lifecycle-manager                 4.6.0   True        False         False      29h
operator-lifecycle-manager-catalog         4.6.0   True        False         False      29h
operator-lifecycle-manager-packageserver   4.6.0   True        False         False      3h59m
service-ca                                 4.6.0   True        False         False      29h
storage                                    4.6.0   True        False         False      4h30m
```

或者，通过以下命令，如果所有集群都可用您会接到通知。它还检索并显示凭证：

```
$ ./openshift-install --dir <installation_directory> wait-for install-complete 1
```

1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

输出示例

```
INFO Waiting up to 30m0s for the cluster to initialize...
```

Cluster Version Operator 完成从 Kubernetes API 服务器部署 OpenShift Container Platform 集群时，命令运行成功。

重要

- 安装程序生成的 Ignition 配置文件包含在 24 小时后过期的证书，然后在过期时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外情况是，您需要手动批准待处理的 **node-bootstrapper** 证书签名请求（CSR）来恢复 kubelet 证书。如需更多信息，请参阅*从过期的 control plane 证书中恢复*的文档。
- 建议您在生成 12 小时后使用 Ignition 配置文件，因为集群安装后 24 小时证书从 16 小时轮转至 22 小时。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中运行证书更新时避免安装失败。

2. 确认 Kubernetes API 服务器正在与 pod 通信。

   a. 要查看所有 pod 的列表，请使用以下命令：

   ```
   $ oc get pods --all-namespaces
   ```

   输出示例

   ```
   NAMESPACE                       NAME                                         READY   STATUS
   RESTARTS   AGE
   openshift-apiserver-operator    openshift-apiserver-operator-85cb746d55-zqhs8   1/1
   ```

```
Running     1          9m
openshift-apiserver            apiserver-67b9g                          1/1     Running     0
3m
openshift-apiserver            apiserver-ljcmx                          1/1     Running     0
1m
openshift-apiserver            apiserver-z25h4                          1/1     Running     0
2m
openshift-authentication-operator authentication-operator-69d5d8bf84-vh2n8        1/1
Running     0          5m
...
```

b. 使用以下命令，查看上一命令的输出中所列 pod 的日志：

```
$ oc logs <pod_name> -n <namespace> 1
```

1 指定 pod 名称和命名空间，如上一命令的输出中所示。

如果 pod 日志显示，Kubernetes API 服务器可以与集群机器通信。

3. 在 Cluster registration 页面注册您的集群。

7.2.14. OpenShift Container Platform 的 Telemetry 访问

在 OpenShift Container Platform 4.6 中，默认运行的 Telemetry 服务提供有关集群健康状况和成功更新的指标，需要访问互联网。如果您的集群连接到互联网，Telemetry 会自动运行，而且集群会注册到 OpenShift Cluster Manager。

确认 OpenShift Cluster Manager 清单正确后，可以由 Telemetry 自动维护，也可以使用 OpenShift Cluster Manager 手动维护，使用订阅监控来跟踪帐户或多集群级别的 OpenShift Container Platform 订阅。

其他资源

- 有关 Telemetry 服务的更多信息，请参阅关于远程健康监控。

7.2.15. 收集调试信息

您可以收集有助于在 IBM Z 中安装 OpenShift Container Platform 时对特定问题进行故障排除和调试的调试信息。

先决条件

- 安装了 **oc** CLI 工具

流程

1. 登录到集群：

```
$ oc login
```

2. 在您要收集硬件信息的节点中，启动一个调试容器：

```
$ oc debug node/<nodename>
```

3. 进入 **/host** 文件系统并启动 **toolbox**:

```
$ chroot /host
$ toolbox
```

4. 收集 **dbginfo** 数据：

```
$ dbginfo.sh
```

5. 然后可以使用 **scp**来获取数据。

其他资源

- 请参阅 如何在没有 SSH 的 OpenShift Container Platform 版本 4 节点中生成 SOSREPORT 。

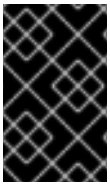
7.2.16. 后续步骤

- 自定义集群。
- 如果您用来安装集群的镜像 registry 具有一个可信任的 CA，通过配置额外的信任存储将其添加到集群中。

第 8 章 在 IBM POWER 系统上安装

8.1. 在 IBM POWER 系统上安装集群

在 OpenShift Container Platform 版本 4.6 中，您可以在您置备的 IBM Power Systems 系统上安装集群。

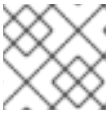


重要

非裸机平台还有其他注意事项。在尝试在此类环境中安装 OpenShift Container Platform 集群前，请参阅有关在未经测试的平台上部署 OpenShift Container Platform 的指南 中的信息。

先决条件

- 在开始安装进程前，您必须清理安装目录。这可保证在安装过程中创建和更新所需的安装文件。
- 为集群置备使用 NFS 的持久性存储。若要部署私有镜像 registry，您的存储必须提供 **ReadWriteMany** 访问模式。
- 查看有关 OpenShift Container Platform 安装和更新 流程的详细信息。
- 如果使用防火墙，则必须将其配置为允许集群需要访问的站点。



注意

如果您要配置代理，请务必也要查看此站点列表。

8.1.1. OpenShift Container Platform 的互联网访问

在 OpenShift Container Platform 4.6 中，您需要访问互联网来安装集群。

您必须具有以下互联网访问权限：

- 访问 OpenShift Cluster Manager 以下载安装程序并执行订阅管理。如果集群可以访问互联网，并且没有禁用 Telemetry，该服务会自动授权您的集群。
- 访问 Quay.io，以获取安装集群所需的软件包。
- 获取执行集群更新所需的软件包。

重要

如果您的集群无法直接访问互联网，则可以在置备的某些类基础架构上执行受限网络安装。在此过程中，您要下载所需的内容，并使用它在镜像 registry（mirror registry） 中填充安装集群并生成安装程序所需的软件包。对于某些安装类型，集群要安装到的环境不需要访问互联网。在更新集群之前，要更新 registry 镜像系统中的内容。

8.1.2. 具有用户置备基础架构的集群的机器要求

对于含有用户置备的基础架构的集群，您必须部署所有所需的机器。

8.1.2.1. 所需的机器

最小的 OpenShift Container Platform 集群需要下列主机：

- 一个临时 bootstrap 机器
- 三台 control plane 或 master 机器
- 至少两台计算机器，也称为 worker 机器。

注意

集群要求 bootstrap 机器在三台 control plane 机器上部署 OpenShift Container Platform 集群。您可在安装集群后删除 bootstrap 机器。

重要

要保持集群的高可用性，请将独立的物理主机用于这些集群机器。

bootstrap 和 control plane 机器必须使用 Red Hat Enterprise Linux CoreOS (RHCOS) 作为操作系统。但是，计算机器可以在 Red Hat Enterprise Linux CoreOS(RHCOS)或 Red Hat Enterprise Linux(RHEL)7.9 之间进行选择。

请注意，RHCOS 基于 Red Hat Enterprise Linux（RHEL） 8，并继承其所有硬件认证和要求。请查看Red Hat Enterprise Linux 技术功能及限制 。

8.1.2.2. 网络连接要求

所有 Red Hat Enterprise Linux CoreOS (RHCOS) 机器在启动过程中需要 **initramfs** 中的网络从 Machine Config Server 获取 Ignition 配置文件。在初次启动过程中，需要一个 DHCP 服务器或设置了静态 IP 地址来建立网络连接，以下载它们的 Ignition 配置文件。另外，集群中的每个 OpenShift Container Platform 节点都必须有权访问网络时间协议（NTP）服务器。如果 DHCP 服务器提供 NTP 服务器信息，Red Hat Enterprise Linux CoreOS（RHCOS）机器上的 chrony 时间服务会读取信息，并可与 NTP 服务器同步时钟。

8.1.2.3. 最低资源要求

每台集群机器都必须满足以下最低要求：

表 8.1. 最低资源要求

机器	操作系统	vCPU [1]	虚拟内存	存储	IOPS [2]
bootstrap	RHCOS	2	16 GB	100 GB	300
Control plane	RHCOS	2	16 GB	100 GB	300
Compute	RHCOS	2	8 GB	100 GB	300

1. 当未启用并发多线程 (SMT) 或超线程时，一个 vCPU 相当于一个物理内核。启用后，使用以下公式来计算对应的比例：（每个内核数的线程）× sockets = vCPU。

2. OpenShift Container Platform 和 Kubernetes 对磁盘性能非常敏感，建议使用更快的存储速度，特别是 control plane 节点上需要 10 ms p99 fsync 持续时间的 etcd。请注意，在许多云平台上，存储大小和 IOPS 可一起扩展，因此您可能需要过度分配存储卷来获取足够的性能。

8.1.2.4. 证书签名请求管理

在使用您置备的基础架构时，集群只能有限地访问自动机器管理，因此您必须提供一种在安装后批准集群证书签名请求 (CSR) 的机制。**kube-controller-manager** 只能批准 kubelet 客户端 CSR。 **machine-approver** 无法保证使用 kubelet 凭证请求的提供证书的有效性，因为它不能确认是正确的机器发出了该请求。您必须决定并实施一种方法，以验证 kubelet 提供证书请求的有效性并进行批准。

8.1.3. 创建用户置备的基础架构

在部署采用用户置备的基础架构的 OpenShift Container Platform 集群前，您必须创建底层基础架构。

先决条件

- 在为集群创建支持基础架构之前，请参阅OpenShift Container Platform 4.x Tested Integrations页。

流程

1. 在每个节点上配置 DHCP 或设置静态 IP 地址。
2. 提供所需的负载均衡器。
3. 配置机器的端口。
4. 配置 DNS。
5. 确保网络可以正常工作。

8.1.3.1. 用户置备的基础架构对网络的要求

所有 Red Hat Enterprise Linux CoreOS（RHCOS）机器在启动过程中需要 **initramfs** 中的网络从机器配置服务器获取 Ignition 配置。

在初次启动过程中，需要一个 DHCP 服务器或集群中的每个机器都设置了静态 IP 地址来建立网络连接，以下载它们的 Ignition 配置文件。

建议您使用 DHCP 服务器为集群进行长期机器管理。确保 DHCP 服务器已配置为向集群机器提供持久 IP 地址和主机名。

Kubernetes API 服务器必须能够解析集群机器的节点名称。如果 API 服务器和 worker 节点位于不同的区域中，您可以配置默认 DNS 搜索区域，以便 API 服务器能够解析节点名称。另一种支持的方法是始终在节点对象和所有 DNS 请求中使用完全限定域名来指代主机。

您必须配置机器间的网络连接，以便集群组件进行通信。每台机器都必须能够解析集群中所有其他机器的主机名。

表 8.2. 所有机器到所有机器

协议	端口	描述

协议	端口	描述
ICMP	N/A	网络可访问性测试
TCP	**1936**	指标
	9000-9999	主机级别的服务，包括端口 **9100-9101** 上的节点导出器和端口 **9099** 上的 Cluster Version Operator。
	10250-10259	Kubernetes 保留的默认端口
	10256	openshift-sdn
UDP	**4789**	VXLAN 和 Geneve
	6081	VXLAN 和 Geneve
	9000-9999	主机级别的服务，包括端口 **9100-9101** 上的节点导出器。
TCP/UDP	**30000-32767**	Kubernetes 节点端口

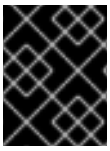
表 8.3. 要通过控制平面的所有机器

协议	端口	描述
TCP	**6443**	Kubernetes API

表 8.4. control plane 机器到 control plane 机器

协议	端口	描述
TCP	**2379-2380**	etcd 服务器和对等端口

网络拓扑要求
您为集群置备的基础架构必须满足下列网络拓扑要求。



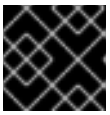
重要

OpenShift Container Platform 要求所有节点都能访问互联网，以便为平台容器提取镜像并向红帽提供遥测数据。

负载均衡器
在安装 OpenShift Container Platform 前，您必须置备两个满足以下要求的负载均衡器：

1. **API 负载均衡器**：提供一个通用端点，供用户（包括人和机器）与平台交互和配置。配置以下条件：

- 只适用于第 4 层负载均衡。这可被称为 Raw TCP、SSL Passthrough 或者 SSL 桥接模式。如果使用 SSL Bridge 模式，必须为 API 路由启用 Server Name Indication（SNI）。
- 无状态负载平衡算法。这些选项根据负载均衡器的实现而有所不同。



重要

不要为 API 负载均衡器配置会话持久性。

在负载均衡器的前端和后台配置以下端口：

表 8.5. API 负载均衡器

端口	后端机器（池成员）	内部	外部	描述
6443	Bootstrap 和 control plane.bootstrap 机器初始化集群 control plane 后，您要从负载均衡器中删除 bootstrap 机器。您必须为 API 服务器健康检查探测配置 **/readyz** 端点。	X	X	Kubernetes API 服务器
22623	Bootstrap 和 control plane.bootstrap 机器初始化集群 control plane 后，您要从负载均衡器中删除 bootstrap 机器。	X		机器配置服务器

注意

负载均衡器必须配置为，从 API 服务器关闭 **/readyz** 端点到从池中删除 API 服务器实例时最多需要 30 秒。在 **/readyz** 返回错误或处于健康状态后的时间范围内，端点必须被删除或添加。每 5 秒或 10 秒探测一次，有两个成功请求处于健康状态，三个成为不健康的请求经过测试。

2. **应用程序入口负载均衡器**:提供来自集群外部的应用程序流量流量的 Ingress 点。配置以下条件：
   - 只适用于第 4 层负载均衡。这可被称为 Raw TCP、SSL Passthrough 或者 SSL 桥接模式。如果使用 SSL Bridge 模式，您必须为 Ingress 路由启用Server Name Indication（SNI）。
   - 建议根据可用选项以及平台上托管的应用程序类型，使用基于连接的或者基于会话的持久性。

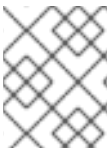
在负载均衡器的前端和后台配置以下端口：

表 8.6. 应用程序入口负载均衡器

端口	后端机器（池成员）	内部	外部	描述
443	默认运行入口路由器 Pod、计算或 worker 的机器。	X	X	HTTPS 流量
80	默认运行入口路由器 Pod、计算或 worker 的机器。	X	X	HTTP 流量

提示

如果负载均衡器可以看到客户端的真实 IP 地址，启用基于 IP 的会话持久性可提高使用端到端 TLS 加密的应用程序的性能。



注意

OpenShift Container Platform 集群需要正确配置入口路由器。control plane 初始化后，您必须配置入口路由器。

NTP 配置

OpenShift Container Platform 集群默认配置为使用公共网络时间协议（NTP）服务器。如果要使用本地企业 NTP 服务器，或者集群部署在断开连接的网络中，您可以将集群配置为使用特定的时间服务器。如需更多信息，请参阅*配置 chrony 时间服务*的文档。

如果 DHCP 服务器提供 NTP 服务器信息，Red Hat Enterprise Linux CoreOS（RHCOS）机器上的 chrony 时间服务会读取信息，并可与 NTP 服务器同步时钟。

其他资源

- 配置 chrony 时间服务

8.1.3.2. 用户置备 DNS 要求

DNS 用于名称解析和反向名称解析。DNS A/AAAA 或 CNAME 记录用于名称解析，PTR 记录用于反向解析名称。反向记录很重要，因为 Red Hat Enterprise Linux CoreOS（RHCOS）使用反向记录为所有节点设置主机名。另外，反向记录用于生成 OpenShift Container Platform 需要操作的证书签名请求（CSR）。

采用用户置备的基础架构的 OpenShift Container Platform 集群需要以下 DNS 记录。在每一记录中，**<cluster_name>** 是集群名称，**<base_domain>** 则是您在 **install-config.yaml** 文件中指定的集群基域。完整的 DNS 记录采用如下格式: **<component>.<cluster_name>.<base_domain>.**。

表 8.7. 所需的 DNS 记录

组件	记录	描述
Kubernetes API	**api.<cluster_name>. <base_domain>.**	添加 DNS A/AAAA 或 CNAME 记录，以及 DNS PTR 记录，以识别 control plane 机器的负载均衡器。这些记录必须由集群外的客户端以及集群中的所有节点解析。
	api-int.<cluster_name>. <base_domain>.	添加 DNS A/AAAA 或 CNAME 记录，以及 DNS PTR 记录，以识别 control plane 机器的负载均衡器。这些记录必须可以从集群中的所有节点解析。  **重要** API 服务器必须能够根据在 Kubernetes 中记录的主机名解析 worker 节点。如果 API 服务器无法解析节点名称，则代理的 API 调用会失败，且您无法从 pod 检索日志。

组件	记录	描述
Routes	***.apps.<cluster_name>. <base_domain>.**	添加通配符 DNS A/AAAA 或 CNAME 记录，指向以运行入口路由器 Pod 的机器（默认为 worker 节点）为目标的负载均衡器。这些记录必须由集群外的客户端以及集群中的所有节点解析。
bootstrap	**bootstrap.<cluster_name>. <base_domain>.**	添加 DNS A/AAAA 或 CNAME 记录，以及 DNS PTR 记录来识别 bootstrap 机器。这些记录必须由集群中的节点解析。
Master 主机	**<master><n>. <cluster_name>. <base_domain>.**	DNS A/AAAA 或 CNAME 记录，以识别 control plane 节点（也称为 master 节点）的每台机器。这些记录必须由集群中的节点解析。
Worker 主机	**<worker><n>. <cluster_name>. <base_domain>.**	添加 DNS A/AAAA 或 CNAME 记录，以识别 worker 节点的每台机器。这些记录必须由集群中的节点解析。

提示

您可以使用 **nslookup <hostname>** 命令来验证名称解析。您可以使用 **dig -x <ip_address>** 命令来验证 PTR 记录的反向名称解析。

下面的 BIND 区文件的例子展示了关于名字解析的 A 记录的例子。这个示例的目的是显示所需的记录。这个示例不是为选择一个名称解析服务提供建议。

例 8.1. DNS 区数据库示例

```
$TTL 1W
@ IN SOA ns1.example.com. root (
   2019070700 ; serial
   3H  ; refresh (3 hours)
   30M  ; retry (30 minutes)
   2W  ; expiry (2 weeks)
   1W )  ; minimum (1 week)
 IN NS ns1.example.com.
 IN MX 10 smtp.example.com.
;
;
ns1 IN A 192.168.1.5
smtp IN A 192.168.1.5
;
helper IN A 192.168.1.5
helper.ocp4 IN A 192.168.1.5
;
; The api identifies the IP of your load balancer.
api.ocp4  IN A 192.168.1.5
api-int.ocp4  IN A 192.168.1.5
;
; The wildcard also identifies the load balancer.
```

```
*.apps.ocp4  IN A 192.168.1.5
;
; Create an entry for the bootstrap host.
bootstrap.ocp4 IN A 192.168.1.96
;
; Create entries for the master hosts.
master0.ocp4  IN A 192.168.1.97
master1.ocp4  IN A 192.168.1.98
master2.ocp4  IN A 192.168.1.99
;
; Create entries for the worker hosts.
worker0.ocp4  IN A 192.168.1.11
worker1.ocp4  IN A 192.168.1.7
;
;EOF
```

下面的 BIND 区文件示例显示了反向名字解析的 PTR 记录示例。

例 8.2. 反向记录的 DNS 区数据库示例

```
$TTL 1W
@ IN SOA ns1.example.com. root (
   2019070700 ; serial
   3H  ; refresh (3 hours)
   30M  ; retry (30 minutes)
   2W  ; expiry (2 weeks)
   1W )  ; minimum (1 week)
 IN NS ns1.example.com.
;
; The syntax is "last octet" and the host must have an FQDN
; with a trailing dot.
97 IN PTR master0.ocp4.example.com.
98 IN PTR master1.ocp4.example.com.
99 IN PTR master2.ocp4.example.com.
;
96 IN PTR bootstrap.ocp4.example.com.
;
5 IN PTR api.ocp4.example.com.
5 IN PTR api-int.ocp4.example.com.
;
11 IN PTR worker0.ocp4.example.com.
7 IN PTR worker1.ocp4.example.com.
;
;EOF
```

8.1.4. 生成 SSH 私钥并将其添加到代理中

如果要在集群上执行安装调试或灾难恢复，则必须为 **ssh-agent** 和安装程序提供 SSH 密钥。您可以使用此密钥访问公共集群中的 bootstrap 机器来排除安装问题。

注意

在生产环境中，您需要进行灾难恢复和调试。

您可以使用此密钥以 **core** 用户身份通过 SSH 连接到 master 节点。在部署集群时，此密钥会添加到 **core** 用户的 **~/.ssh/authorized_keys** 列表中。

注意

您必须使用一个本地密钥，而不要使用在特定平台上配置的密钥，如 AWS 密钥对。

流程

1. 如果还没有为计算机上免密码身份验证而配置的 SSH 密钥，请创建一个。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ ssh-keygen -t ed25519 -N '' \
  -f <path>/<file_name> 1
```

1 指定新 SSH 密钥的路径和文件名，如 **~/.ssh/id_rsa**。如果您已有密钥对，请确保您的公钥位于 **~/.ssh** 目录中。

运行此命令会在指定的位置生成不需要密码的 SSH 密钥。

注意

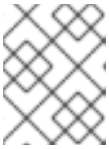
如果您计划在 **x86_64** 架构中安装使用 FIPS 验证的/Modules in Process 加密库的 OpenShift Container Platform 集群，不要创建使用 **ed25519** 算法的密钥。反之，创建一个使用 **rsa** 或 **ecdsa** 算法的密钥。

2. 作为后台任务启动 **ssh-agent** 进程：

```
$ eval "$(ssh-agent -s)"
```

输出示例

```
Agent pid 31874
```



注意

如果您的集群采用 FIPS 模式，则只使用 FIPS 兼容算法来生成 SSH 密钥。密钥必须是 RSA 或 ECDSA。

3. 将 SSH 私钥添加到 **ssh-agent**：

```
$ ssh-add <path>/<file_name> 1
```

输出示例

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

1 指定 SSH 私钥的路径和文件名，如 **~/.ssh/id_rsa**

后续步骤

- 在安装 OpenShift Container Platform 时，为安装程序提供 SSH 公钥。

8.1.5. 获取安装程序

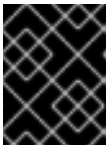
在安装 OpenShift Container Platform 之前，将安装文件下载到本地计算机上。

先决条件

- 运行 Linux 或 macOS 的计算机，本地磁盘空间为 500 MB

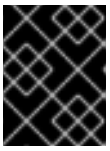
流程

1. 访问 OpenShift Cluster Manager 站点的 Infrastructure Provider 页面。如果您有红帽帐号，请使用自己的凭证登录。如果没有，请创建一个帐户。
2. 选择您的基础架构供应商。
3. 进入适用于您的安装类型的页面，下载您的操作系统的安装程序，并将文件放在要保存安装配置文件的目录中。。



重要

安装程序会在用来安装集群的计算机上创建若干文件。在完成集群安装后，您必须保留安装程序和安装程序所创建的文件。这两个文件都需要删除集群。



重要

删除安装程序创建的文件不会删除您的集群，即使集群在安装过程中失败也是如此。要删除集群，为特定云供应商完成 OpenShift Container Platform 卸载流程。

4. 提取安装程序。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ tar xvf openshift-install-linux.tar.gz
```

5. 从 Red Hat OpenShift Cluster Manager 下载安装 pull secret 。通过此 pull secret，您可以进行所含授权机构提供的服务的身份验证，这些服务包括为 OpenShift Container Platform 组件提供容器镜像的 Quay.io。

8.1.6. 通过下载二进制文件安装 OpenShift CLI

您需要安装 CLI（**oc**） 来使用命令行界面与 OpenShift Container Platform 进行交互。您可在 Linux 、Windows 或 macOS 上安装 **oc**。

重要

如果安装了旧版本的 **oc**，则无法使用 OpenShift Container Platform 4.6 中的所有命令。下载并安装新版本的 **oc**。

8.1.6.1. 在 Linux 上安装 OpenShift CLI

您可以按照以下流程在 Linux 上安装 OpenShift CLI（**oc**）二进制文件。

流程

1. 进入到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 在 **Version** 下拉菜单中选择相应的版本。
3. 单击 **OpenShift v4.6 Linux 客户端** 条目旁边的 **Download Now**，再保存文件。
4. 解包存档：

```
$ tar xvzf <file>
```

5. 把 **oc** 二进制代码放到 **PATH** 中的目录中。
   执行以下命令可以查看当前的 **PATH** 设置：

```
$ echo $PATH
```

安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
$ oc <command>
```

8.1.6.2. 在 Windows 上安装 OpenShift CLI

您可以按照以下流程在 Windows 上安装 OpenShift CLI（**oc**）二进制代码。

流程

1. 进入到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 在 **Version** 下拉菜单中选择相应的版本。
3. 单击 **OpenShift v4.6 Windows 客户端**条目旁边的 **Download Now**，再保存文件。
4. 使用 ZIP 程序解压存档。
5. 把 **oc** 二进制代码放到 **PATH** 中的目录中。
   要查看您的 **PATH**，请打开命令提示窗口并执行以下命令：

```
C:\> path
```

安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
C:\> oc <command>
```

8.1.6.3. 在 macOS 上安装 OpenShift CLI

您可以按照以下流程在 macOS 上安装 OpenShift CLI（**oc**）二进制代码。

流程

1. 进入到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 在 **Version** 下拉菜单中选择相应的版本。
3. 单击 **OpenShift v4.6 MacOSX 客户端**条目旁边的 **Download Now**，再保存文件。
4. 解包和解压存档。
5. 将 **oc** 二进制文件移到 PATH 的目录中。
   要查看您的 **PATH**，打开一个终端窗口并执行以下命令：

```
$ echo $PATH
```

安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
$ oc <command>
```

8.1.7. 手动创建安装配置文件

对于使用用户置备的基础架构的 OpenShift Container Platform 安装，您必须手动生成安装配置文件。

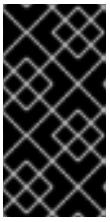
先决条件

- 获取 OpenShift Container Platform 安装程序和集群的访问令牌。

流程

1. 创建用来存储您所需的安装资产的安装目录：

```
$ mkdir <installation_directory>
```



重要

您必须创建目录。一些安装信息，如 bootstrap X.509 证书，有较短的过期间隔，因此不要重复使用安装目录。如果要重复使用另一个集群安装中的个别文件，可以将其复制到您的目录中。但是，一些安装数据的文件名可能会在发行版本之间有所改变。从 OpenShift Container Platform 老版本中复制安装文件时要格外小心。

2. 自定义以下 **install-config.yaml** 文件模板，并将它保存到 **<installation_directory>** 中。

注意

此配置文件必须命名为 **install-config.yaml**。

3. 备份 **install-config.yaml** 文件，以便用于安装多个集群。

重要

install-config.yaml 文件会在安装过程的下一步骤中消耗掉。现在必须备份它。

8.1.7.1. IBM Power 系统的 install-config.yaml 文件示例

您可以自定义 **install-config.yaml** 文件，以指定有关 OpenShift Container Platform 集群平台的更多信息，或修改所需参数的值。

```
apiVersion: v1
baseDomain: example.com (1)
compute: (2)
- hyperthreading: Enabled (3)
  name: worker
  replicas: 0 (4)
  architecture : ppc64le
controlPlane: (5)
  hyperthreading: Enabled (6)
  name: master
  replicas: 3 (7)
  architecture : ppc64le
metadata:
  name: test (8)
networking:
  clusterNetwork:
  - cidr: 10.128.0.0/14 (9)
    hostPrefix: 23 (10)
  networkType: OpenShiftSDN
  serviceNetwork: (11)
  - 172.30.0.0/16
platform:
  none: {} (12)
fips: false (13)
pullSecret: '{"auths": ...}' (14)
sshKey: 'ssh-ed25519 AAAA...' (15)
```

(1) 集群的基域。所有 DNS 记录都必须是这个基域的子域，并包含集群名称。

(2) (5) **controlPlane** 部分是一个单映射，但 **compute** 部分是一系列映射。为满足不同数据结构的要求，**compute** 部分的第一行必须以连字符 **-** 开头，**controlPlane** 部分的第一行则不可以连字符开头。只使用一个 control plane 池。

(3) (6) 是否要启用或禁用并发多线程（SMT）或**超线程**。默认情况下，启用 SMT 可提高机器内核的性能。您可以通过将参数值设为 **Disabled** 来禁用。如果禁用 SMT，则必须在所有集群机器中禁用它，其中包括 control plane 和计算机器。

注意

默认启用并发多线程（SMT）。如果在 BIOS 设置中没有启用 SMT，**hyperthreading** 参数不会起作用。

重要

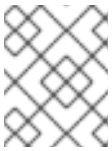
如果您禁用 **hyperthreading**（无论是在 BIOS 中还是在 **install-config.yaml** 中），请确保您对可能会造成的机器性能显著降低的情况有所考虑。

4 **replicas** 参数的值必须设置为 **0**。此参数控制集群为您创建和管理的 worker 数量，使用用户置备的基础架构时集群不会执行这些功能。在完成 OpenShift Container Platform 安装前，您必须手动为集

7 您添加到集群的 control plane 机器数量。由于集群将这个值用作集群中 etcd 端点的数量，因此该值必须与您部署的 control plane 机器数量匹配。

8 您在 DNS 记录中指定的集群名称。

9 从中分配 pod IP 地址的 IP 地址块。此块不得与现有的物理网络重叠。这些 IP 地址用于 pod 网络。如果您需要从外部网络访问 pod，请配置负载均衡器和路由器来管理流量。



注意

类 E CIDR 范围保留给以后使用。要使用 Class E CIDR 范围，您必须确保您的网络环境接受 Class E CIDR 范围内的 IP 地址。

10 分配给每个单独节点的子网前缀长度。例如，如果 **hostPrefix** 设为 **23**，则每个节点从所给的 **cidr** 中分配一个 **/23** 子网，这样就能有 510 (2^(32 - 23) - 2) 个 Pod IP 地址。如果您需要从外部网络访问节点，请配置负载均衡器和路由器来管理流量。

11 用于服务 IP 地址的 IP 地址池。您只能输入一个 IP 地址池。此块不得与现有的物理网络重叠。如果您需要从外部网络访问服务，请配置负载均衡器和路由器来管理流量。

12 您必须将平台设置为 **none**。您无法为 IBM Power 系统基础架构提供额外的平台配置变量。

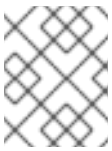
13 是否启用或禁用 FIPS 模式。默认情况下不启用 FIPS 模式。如果启用了 FIPS 模式，运行 OpenShift Container Platform 的 Red Hat Enterprise Linux CoreOS (RHCOS) 机器会绕过默认的 Kubernetes 加密套件，并使用由 RHCOS 提供的加密模块。

重要

只有在 **x86_64** 架构中的 OpenShift Container Platform 部署支持 FIPS 验证的/Modules in Process 加密库。

14 Red Hat OpenShift Cluster Manager 中的 pull secret 。通过此 pull secret，您可以进行所含授权机构提供的服务的身份验证，这些服务包括为 OpenShift Container Platform 组件提供容器镜像的 Quay.io。

15 Red Hat Enterprise Linux CoreOS (RHCOS) 中 **core** 用户的默认 SSH 密钥的公钥部分。



注意

对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 **ssh-agent** 进程使用的 SSH 密钥。

8.1.7.2. 在安装过程中配置集群范围代理

生产环境可能会拒绝直接访问互联网，而是提供 HTTP 或 HTTPS 代理。您可以通过在 **install-config.yaml** 文件中配置代理设置，将新的 OpenShift Container Platform 集群配置为使用代理。

先决条件

- 您有一个现有的 **install-config.yaml** 文件。

- 您检查了集群需要访问的站点，并决定是否需要绕过代理。默认情况下代理所有集群出口流量，包括对托管云供应商 API 的调用。您需要将站点添加到 **Proxy** 对象的 **spec.noProxy** 字段来绕过代理。

注意

Proxy 对象 **status.noProxy** 字段使用安装配置中的 **networking.machineNetwork[].cidr**、**networking.clusterNetwork[].cidr** 和 **networking.serviceNetwork[]** 字段的值填充。

对于在 Amazon Web Services(AWS)、Google Cloud Platform(GCP)、Microsoft Azure 和 Red Hat OpenStack Platform(RHOSP)上安装，**Proxy** 对象 **status.noProxy** 字段也会使用实例元数据端点填充(**169.254.169.254**)。

流程

1. 编辑 **install-config.yaml** 文件并添加代理设置。例如：

```
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: example.com 3
additionalTrustBundle: | 4
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
...
```

1. 用于创建集群外 HTTP 连接的代理 URL。URL 必须是 **http**。
2. 用于创建集群外 HTTPS 连接的代理 URL。
3. 要排除在代理中的目标域名、IP 地址或其他网络 CIDR 的逗号分隔列表。在域前面加 **.** 来仅匹配子域。例如：**.y.com** 匹配 **x.y.com**，但不匹配 **y.com**。使用 ***** 绕过所有目的地的代理。
4. 如果提供，安装程序会在 **openshift-config 命名空间中生成名为 user-ca- bundle** 的配置映射来保存额外的 CA 证书。如果您提供 **additionalTrustBundle** 和至少一个代理设置，则 **Proxy** 对象会被配置为引用 **trustedCA** 字段中的 **user-ca-bundle** 配置映射。然后，Cluster Network Operator 会创建一个 **trusted-ca-bundle** 配置映射，该配置映射将为 **trustedCA** 参数指定的内容与 RHCOS 信任捆绑包合并。**additionalTrustBundle** 字段是必需的，除非代理的身份证书由来自 RHCOS 信任捆绑包的颁发机构签名。

注意

安装程序不支持代理的 **readinessEndpoints** 字段。

2. 保存该文件，并在安装 OpenShift Container Platform 时引用。

安装程序会创建一个名为 **cluster** 的集群范围代理，该代理使用提供的 **install-config.yaml** 文件中的代理设置。如果没有提供代理设置，仍然会创建一个 **cluster Proxy** 对象，但它会有一个空 **spec**。

注意

只支持名为 **cluster** 的 **Proxy** 对象，且无法创建额外的代理。

8.1.8. 创建 Kubernetes 清单和 Ignition 配置文件

由于您必须修改一些集群定义文件并要手动启动集群机器，因此您必须生成 Kubernetes 清单和 Ignition 配置文件，集群需要这两项来创建其机器。

安装配置文件转换为 Kubernetes 清单。清单嵌套到 Ignition 配置文件中，稍后用于创建集群。

重要

- 安装程序生成的 Ignition 配置文件包含在 24 小时后过期的证书，然后在过期时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外情况是，您需要手动批准待处理的 **node-bootstrapper** 证书签名请求（CSR）来恢复 kubelet 证书。如需更多信息，请参阅*从过期的 control plane 证书中恢复*的文档。
- 建议您在生成 12 小时后使用 Ignition 配置文件，因为集群安装后 24 小时证书从 16 小时轮转至 22 小时。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中运行证书更新时避免安装失败。

先决条件

- 已获得 OpenShift Container Platform 安装程序。
- 已创建 **install-config.yaml** 安装配置文件。

流程

1. 切换到包含安装程序的目录，并为集群生成 Kubernetes 清单：

```
$ ./openshift-install create manifests --dir <installation_directory> 1
```

1 对于 **<installation_directory>**，请指定含有您创建的 **install-config.yaml** 文件的安装目录。

2. 检查 **<installation_directory>/manifests/cluster-scheduler-02-config.yml** Kubernetes 清单文件中的 **mastersSchedulable** 参数是否已设置为 **false**。此设置可防止在 control plane 机器上调度 pod:
   a. 打开 **<installation_directory>/manifests/cluster-scheduler-02-config.yml** 文件。
   b. 找到 **mastersSchedulable** 参数并确保它被设置为 **false**。
   c. 保存并退出文件。
3. 要创建 Ignition 配置文件，从包含安装程序的目录运行以下命令：

```
$ ./openshift-install create ignition-configs --dir <installation_directory> 1
```

1 对于 **<installation_directory>**，请指定相同的安装目录。

该目录中将生成以下文件：

```
.
├── auth
│   ├── kubeadmin-password
│   └── kubeconfig
├── bootstrap.ign
├── master.ign
├── metadata.json
└── worker.ign
```

8.1.9. 创建 Red Hat Enterprise Linux CoreOS (RHCOS) 机器

在您置备的 IBM Power Systems 基础架构上安装集群前，必须先创建 RHCOS 机器供它使用。按照相应的步骤，使用 ISO 镜像或网络 PXE 启动来创建机器。

8.1.9.1. 使用 ISO 镜像创建 Red Hat Enterprise Linux CoreOS (RHCOS) 机器

在您置备的 IBM Power Systems 基础架构上安装集群前，必须先创建 RHCOS 机器供它使用。您可以使用 ISO 镜像来创建这些机器。

先决条件

- 获取集群的 Ignition 配置文件。
- 具有可从计算机以及您创建的机器访问的 HTTP 服务器的访问权限。

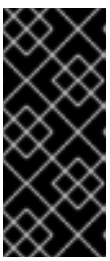
流程

1. 将安装程序创建的 control plane、计算和 bootstrap Ignition 配置文件上传到 HTTP 服务器。记下这些文件的 URL。

重要

如果您计划在安装完成后在集群中添加更多计算机器，请不要删除这些文件。

2. 从 RHCOS 镜像镜像页面获取您选择的操作系统实例安装方法所需的 RHCOS 镜像。



重要

RHCOS 镜像可能不会随着 OpenShift Container Platform 的每一发行版本都有改变。您必须下载最高版本的镜像，其版本号应小于或等于您安装的 OpenShift Container Platform 版本。如果可用，请使用与 OpenShift Container Platform 版本匹配的镜像版本。此流程只使用 ISO 镜像。此安装类型不支持 RHCOS qcow2 镜像。

ISO 文件名类似以下示例：

rhcos-<version>-live.<architecture>.iso

3. 使用 ISO 启动 RHCOS 安装。使用如下安装选项之一：
   - 将 ISO 镜像刻录到磁盘并直接启动。

- 通过 LOM 接口使用 ISO 重定向。

4. 引导 ISO 镜像。您可以中断安装引导过程来添加内核参数。然而，在这个 ISO 过程中，您应该使用 **coreos-installer** 命令而不是添加内核参数。如果您在没有选项或中断的情况下运行 live 安装程序，安装程序将引导至 live 系统上的 shell 提示符，准备好将 RHCOS 安装到磁盘中。

5. 在运行 **coreos-installer** 前，请参阅 *高级 RHCOS 安装参考* 部分，以了解配置功能的不同方法，如网络和磁盘分区。

6. 运行 **coreos-installer** 命令。您至少必须识别节点类型的 Ignition 配置文件位置，以及您要安装到的磁盘位置。下面是一个示例：

```
$ sudo coreos-installer install \
    --ignition-url=https://host/worker.ign /dev/sda
```

7. 安装 RHCOS 后，系统会重启。系统重启过程中，它会应用您指定的 Ignition 配置文件。

8. 继续为集群创建其他机器。

重要

此刻您必须创建 bootstrap 和 control plane 机器。如果 control plane 机器不可调度（这是默认调度），则在安装集群前至少会创建两台计算机器。

8.1.9.1.1. 高级 RHCOS 安装参考

本节演示了网络配置和其他高级选项，允许您修改 Red Hat Enterprise Linux CoreOS（RHCOS）手动安装过程。下表描述了您可以与 RHCOS live installer 和 **coreos-installer** 命令一起使用的内核参数和命令行选项。

RHCOS 启动提示下的路由和绑定选项

如果从 ISO 镜像安装 RHCOS，您可以在引导该镜像时手动添加内核参数以配置节点的网络。如果没有使用网络参数，则安装默认为使用 DHCP。

重要

添加网络参数时，还必须添加 **rd.neednet=1** 内核参数。

下表描述了如何为实时 ISO 安装使用 **ip=** 、 **nameserver=** 和 **bond=** 内核参数。

注意

在添加内核参数时顺序非常重要： **ip=**，**nameserver=**，然后 **bond=**。

ISO 的路由和绑定选项

下表提供了配置 Red Hat Enterprise Linux CoreOS（RHCOS）节点网络的示例。这些是在系统引导过程中传递给 **dracut** 工具的网络选项。有关 **dracut** 支持的网络选项的详情，请参考 **dracut.cmdline** 手册页。

描述	例子
要配置一个 IP 地址，可以使用 DHCP(**ip=dhcp**)或者设置单独的静态 IP 地址(**ip=<host_ip>**)。然后在每个节点上指定 DNS 服务器 IP 地址(**nameserver=<dns_ip>**)。这个示例设置： • 节点的 IP 地址为 **10.10.10.2** • 网关地址为 **10.10.10.254** • 子网掩码为 **255.255.255.0** • 主机名为 **core0.example.com** • DNS 服务器地址为 **4.4.4.41**	ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp1s0:none nameserver=4.4.4.41
通过指定多个 **ip=** 条目来指定多个网络接口。	ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp1s0:none ip=10.10.10.3::10.10.10.254:255.255.255.0:core0.example.com:enp2s0:none
可选：您可以通过设置一个 **rd.route=** 值来配置到额外网络的路由。 如果额外网络网关与主要网络网关不同，则默认网关必须是主要网络网关。	配置默认网关： ip=::10.10.10.254:::: 为额外网络配置路由： rd.route=20.20.20.0/24:20.20.20.254:enp2s0
在单一接口中禁用 DHCP，比如当有两个或者多个网络接口时，且只有一个接口被使用。	ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp1s0:none ip=::::core0.example.com:enp2s0:none
您可以将系统中 DHCP 和静态 IP 配置与多个网络接口结合在一起。	ip=enp1s0:dhcp ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp2s0:none

描述	例子
可选： 您可以使用 **vlan=** 参数在单独的接口上配置 VLAN。	在网络接口中配置 VLAN 并使用静态 IP 地址： ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp2s0.100:none vlan=enp2s0.100:enp2s0 在网络接口中配置 VLAN 并使用 DHCP： ip=enp2s0.100:dhcp vlan=enp2s0.100:enp2s0
您可以为每个服务器添加一个 **nameserver=** 条目来提供多个 DNS 服务器。	nameserver=1.1.1.1 nameserver=8.8.8.8
可选：使用 **bond=** 选项支持将多个网络接口绑定到一个接口。在这两个示例中： - 配置绑定接口的语法为：**bond=name[:network_interfaces][:options]** - *name* 是绑定设备名称（**bond0**），*network_interfaces* 代表用逗号分开的物理（以太网）接口(**em1,em2**)的列表，*options* 是用逗号分开的绑定选项列表。输入 **modinfo bonding** 查看可用选项。 - 当使用 **bond=** 创建绑定接口时，您必须指定如何分配 IP 地址以及绑定接口的其他信息。	要将绑定的接口配置为使用 DHCP，请将绑定的 IP 地址设置为 **dhcp**。例如： bond=bond0:em1,em2:mode=active-backup ip=bond0:dhcp 要将绑定接口配置为使用静态 IP 地址，请输入您需要的特定 IP 地址以及相关信息。例如： bond=bond0:em1,em2:mode=active-backup ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:bond0:none
可选： 您可以使用 **vlan=** 参数在绑定接口上配置 VLAN。	使用 VLAN 配置绑定接口并使用 DHCP： ip=bond0.100:dhcp bond=bond0:em1,em2:mode=active-backup vlan=bond0.100:bond0 使用 VLAN 配置绑定接口，并使用静态 IP 地址： ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:bond0.100:none bond=bond0:em1,em2:mode=active-backup vlan=bond0.100:bond0

描述	例子
可选：通过使用 **team=** 参数，网络合作可用作绑定的替代选择。在此例中： • 配置组接口的语法为：**team=name[:network_interfaces]** *name* 是组设备名称(**team0**)，**network** *_interfaces* 代表以逗号分隔的物理（以太网）接口（**em1**、**em2**）列表。  **注意** 当 RHCOS 切换到即将推出的 RHEL 版本时，团队计划会被弃用。如需更多信息，请参阅红帽知识库文章。	配置网络团队： team=team0:em1,em2 ip=team0:dhcp

8.1.9.2. 通过 PXE 或 iPXE 启动来创建 Red Hat Enterprise Linux CoreOS (RHCOS) 机器

在安装使用手动置备 RHCOS 节点（如裸机）的集群前，您必须创建 RHCOS 机器供其使用。您可以使用 PXE 或 iPXE 启动来创建机器。

先决条件

- 获取集群的 Ignition 配置文件。
- 配置合适的 PXE 或 iPXE 基础架构。
- 具有 HTTP 服务器的访问权限，以便您可从计算机进行访问。

流程

1. 将安装程序创建的 master、worker 和 bootstrap Ignition 配置文件上传到 HTTP 服务器。记下这些文件的 URL。

重要

您可以在 Ignition 配置中添加或更改配置设置，然后将其保存到 HTTP 服务器。如果您计划在安装完成后在集群中添加更多计算机器，请不要删除这些文件。

2. 从 RHCOS 镜像 镜像页面获取 RHCOS **内核、initram fs 和 rootfs** 文件。

重要

RHCOS 工件（artifact）可能不会随着 OpenShift Container Platform 的每个发行版本而改变。您必须下载最高版本的工件，其版本号应小于或等于您安装的 OpenShift Container Platform 版本。这个过程只使用下面描述的正确 **kernel**、**initramfs** 和 **rootfs** 工件。此安装类型不支持 RHCOS qcow2 镜像。

文件名包含 OpenShift Container Platform 版本号。它们类似以下示例：

- **kernel**: **rhcos-<version>-live-kernel-<architecture>**

- **initramfs**: **rhcos-<version>-live-initramfs.<architecture>.img**
- **rootfs**: **rhcos-<version>-live-rootfs.<architecture>.img**

3. 上传引导方法所需的额外文件：
   - 对于传统的 PXE，将 **kernel** 和 **initramfs** 文件上传到 TFTP 服务器，并将 **rootfs** 文件上传到 HTTP 服务器。
   - 对于 iPXE，将 **kernel**、**initram** fs 和 **rootfs** 文件上传到 HTTP 服务器。

重要

如果您计划在安装完成后在集群中添加更多计算机器，请不要删除这些文件。

4. 配置网络启动基础架构，以便在安装 RHCOS 后机器可从本地磁盘启动。
5. 为 RHCOS 镜像配置 PXE 或 iPXE 安装。
   针对您的环境修改以下示例菜单条目之一，并验证能否正确访问镜像和 Ignition 文件：
   - 对于 PXE：

```
DEFAULT pxeboot
TIMEOUT 20
PROMPT 0
LABEL pxeboot
    KERNEL http://<HTTP_server>/rhcos-<version>-live-kernel-<architecture> (1)
    APPEND initrd=http://<HTTP_server>/rhcos-<version>-live-initramfs.
<architecture>.img coreos.live.rootfs_url=http://<HTTP_server>/rhcos-<version>-live-
rootfs.<architecture>.img coreos.inst.install_dev=/dev/sda
coreos.inst.ignition_url=http://<HTTP_server>/bootstrap.ign (2) (3)
```

(1) 指定上传到 HTTP 服务器的 live **kernel** 文件位置。URL 必须是 HTTP、TFTP 或者 FTP；不支持 HTTPS 和 NFS。

(2) 如果您使用多个 NIC，请在 **ip** 选项中指定一个接口。例如，要在名为 **eno1** 的 NIC 上使用 DHCP，请设置 **ip=eno1:dhcp**。

(3) 指定上传到 HTTP 服务器的 RHCOS 文件的位置。**initrd** 参数值是 **initramfs** 文件的位置，**coreos.live.rootfs_url** 参数值是 **rootfs** 文件的位置，**coreos.inst.ignition_url** 参数值是 bootstrap Ignition 配置文件的位置。您还可以在 **APPEND** 行中添加更多内核参数来配置联网或其他引导选项。

注意

这个配置不会在使用图形控制台的机器上启用串口控制台访问。要配置不同的控制台，请在 **APPEND** 行中添加一个或多个 **console=** 参数。例如，添加 **console=tty0 console=ttyS0** 将第一个 PC 串口设置为主控制台，图形控制台作为二级控制台。如需更多信息，请参阅如何在 Red Hat Enterprise Linux 中设置串行终端和（或）控制台？

   - 对于 iPXE：

```
kernel http://<HTTP_server>/rhcos-<version>-live-kernel-<architecture> initrd=main
```

```
coreos.live.rootfs_url=http://<HTTP_server>/rhcos-<version>-live-rootfs.
<architecture>.img coreos.inst.install_dev=/dev/sda
coreos.inst.ignition_url=http://<HTTP_server>/bootstrap.ign 1 2
initrd --name main http://<HTTP_server>/rhcos-<version>-live-initramfs.
<architecture>.img 3
boot
```

1. 指定上传到 HTTP 服务器的 RHCOS 文件的位置。**kernel** 参数值是 **kernel** 文件的位置，在 UEFI 系统中引导时需要 **initrd=main** 参数。 **coreos.live.rootfs_url** 参数值是 **rootfs** 文件的位置， **coreos.inst.ignition_url** 参数值则是 bootstrap Ignition 配置文件的位置。
2. 如果您使用多个 NIC，请在 **ip** 选项中指定一个接口。例如，要在名为 **eno1** 的 NIC 上使用 DHCP，请设置 **ip=eno1:dhcp**。
3. 指定上传到 HTTP 服务器的 **initramfs** 文件的位置。

注意

这个配置不会在使用图形控制台的机器上启用串口控制台访问。要配置不同的控制台，请在 **kerne** 行中添加一个或多个 **console=** 参数。例如，添加 **console=tty0 console=ttyS0** 将第一个 PC 串口设置为主控制台，图形控制台作为二级控制台。如需更多信息，请参阅如何在 Red Hat Enterprise Linux 中设置串行终端和（或）控制台？

6. 如果使用 PXE UEFI，请执行以下操作：
   a. 提供引导系统所需的 **shim x64.efi 和 grubx64 .efi EFI 二进制文件以及 grub.cfg** 文件。
      - 通过将 RHCOS ISO 挂载到主机，然后将 **images/efiboot.img** 文件挂载到您的主机来提取所需的 EFI 二进制文件：

```
$ mkdir -p /mnt/iso
```

```
$ mkdir -p /mnt/efiboot
```

```
$ mount -o loop rhcos-installer.x86_64.iso /mnt/iso
```

```
$ mount -o loop,ro /mnt/iso/images/efiboot.img /mnt/efiboot
```

      - 从 **efiboot.img** 挂载点，将 **EFI/redhat/shimx64.efi** 和 **EFI/redhat/grubx64.efi** 文件复制到 TFTP 服务器中：

```
$ cp /mnt/efiboot/EFI/redhat/shimx64.efi .
```

```
$ cp /mnt/efiboot/EFI/redhat/grubx64.efi .
```

```
$ umount /mnt/efiboot
```

```
$ umount /mnt/iso
```

- 将 RHCOS ISO 中包含的 **EFI/redhat/grub.cfg** 文件复制到您的 TFTP 服务器中。

b. 编辑 **grub.cfg** 文件使其包含类似如下的参数：

```
menuentry 'Install Red Hat Enterprise Linux CoreOS' --class fedora --class gnu-linux --
class gnu --class os {
 linuxefi rhcos-<version>-live-kernel-<architecture> coreos.inst.install_dev=/dev/sda
coreos.live.rootfs_url=http://<HTTP_server>/rhcos-<version>-live-rootfs.
<architecture>.img coreos.inst.ignition_url=http://<HTTP_server>/bootstrap.ign
 initrdefi rhcos-<version>-live-initramfs.<architecture>.img
}
```

其中：

rhcos-<version>-live-kernel-<architecture>

指定上传到 TFTP 服务器的 **内核** 文件。

http://<HTTP_server>/rhcos-<version>-live-rootfs.<architecture>.img

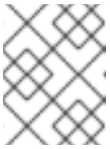
指定上传到 HTTP 服务器的 live rootfs 镜像的位置。

http://<HTTP_server>/bootstrap.ign

指定上传到 HTTP 服务器的 bootstrap Ignition 配置文件的位置。

rhcos-<version>-live-initramfs.<architecture>.img

指定上传到 TFTP 服务器的 **initramfs** 文件的位置。



注意

有关如何为 UEFI 引导配置 PXE 服务器的更多信息，请参阅红帽知识库文章：如何为 Red Hat Enterprise Linux 的 UEFI 引导配置/设置 PXE 服务器？

7. 继续为集群创建机器。

重要

此刻您必须创建 bootstrap 和 control plane 机器。如果 control plane 机器不可调度（这是默认调度），则在安装集群前至少会创建两台计算机器。

8.1.10. 创建集群

要创建 OpenShift Container Platform 集群，请等待您通过安装程序生成的 Ignition 配置文件所置备的机器上完成 bootstrap 过程。

先决条件

- 为集群创建所需的基础架构。
- 已获得安装程序并为集群生成了 Ignition 配置文件。
- 已使用 Ignition 配置文件为集群创建 RHCOS 机器。
- 您的机器可直接访问互联网，或者可以使用 HTTP 或 HTTPS 代理。

流程

1. 监控 bootstrap 过程：

```
$ ./openshift-install --dir <installation_directory> wait-for bootstrap-complete \ 1
    --log-level=info 2
```

1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

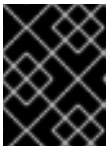
2 要查看不同的安装详情，请指定 **warn**、**debug** 或 **error**，而不要指定 **info**。

输出示例

```
INFO Waiting up to 30m0s for the Kubernetes API at https://api.test.example.com:6443...
INFO API v1.19.0 up
INFO Waiting up to 30m0s for bootstrapping to complete...
INFO It is now safe to remove the bootstrap resources
```

Kubernetes API 服务器提示已在 control plane 机器上完成 bootstrap 时，命令运行成功。

2. bootstrap 过程完成后，请从负载均衡器中删除 bootstrap 机器。



重要

此时您必须从负载均衡器中删除 bootstrap 机器。您还可以删除或重新格式化机器本身。

8.1.11. 使用 CLI 登录到集群

您可以通过导出集群 **kubeconfig** 文件，以默认系统用户身份登录集群。**kubeconfig** 文件包含关于集群的信息，供 CLI 用于将客户端连接到正确集群和 API 服务器。该文件特只适用于一个特定的集群，在 OpenShift Container Platform 安装过程中创建。

先决条件

- 已部署了 OpenShift Container Platform 集群。
- 已安装 **oc** CLI。

流程

1. 导出 **kubeadmin** 凭证：

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

2. 使用导出的配置，验证能否成功运行 **oc** 命令：

```
$ oc whoami
```

输出示例

```
system:admin
```

8.1.12. 批准机器的证书签名请求

将机器添加到集群时，会为您添加的每台机器生成两个待处理证书签名请求（CSR）。您必须确认这些 CSR 已获得批准，或根据需要自行批准。客户端请求必须首先被批准，然后是服务器请求。

先决条件

- 您已将机器添加到集群中。

流程

1. 确认集群可以识别这些机器：

```
$ oc get nodes
```

输出示例

```
NAME     STATUS   ROLES   AGE  VERSION
master-0 Ready    master  63m  v1.19.0
master-1 Ready    master  63m  v1.19.0
master-2 Ready    master  64m  v1.19.0
```

输出将列出您创建的所有机器。

注意

在一些 CSR 被批准前，以上输出可能不包括计算节点（也称为 worker 节点）。

2. 检查待处理的 CSR，并确保可以看到添加到集群中的每台机器都有 **Pending** 或 **Approved** 状态的客户端请求：

```
$ oc get csr
```

输出示例

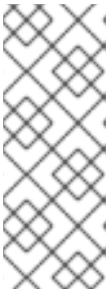
```
NAME        AGE    REQUESTOR                                                   CONDITION
csr-8b2br   15m    system:serviceaccount:openshift-machine-config-operator:node-
bootstrapper   Pending
csr-8vnps   15m    system:serviceaccount:openshift-machine-config-operator:node-
bootstrapper   Pending
...
```

在本例中，两台机器加入了集群。您可能会在列表中看到更多已批准的 CSR。

3. 如果 CSR 没有获得批准，请在所添加机器的所有待处理 CSR 都处于 **Pending** 状态后，为您的集群机器批准这些 CSR：

注意

由于 CSR 会自动轮转，因此请在将机器添加到集群后一小时内批准您的 CSR。如果没有在一小时内批准，证书将会轮转，每个节点将会存在多个证书。您必须批准所有这些证书。批准客户端 CSR 后， Kubelet 为服务证书创建辅助 CSR，这需要手动批准。然后，如果 Kubelet 请求具有相同参数的新证书，则 **machine-approver** 会自动批准后续服务证书续订请求。



注意

对于在未启用机器 API 的平台中运行的集群，如裸机和其他用户置备的基础架构，必须采用一种方法自动批准 kubelet 提供证书请求（CSR）。如果没有批准请求，则 **oc exec**、**oc rsh** 和 **oc logs** 命令将无法成功，因为 API 服务器连接到 kubelet 时需要服务证书。与 Kubelet 端点联系的任何操作都需要此证书批准。这个方法必须监视新的 CSR，确认 CSR 由 **system:node** 或 **system:admin** 组中的 **node-bootstrapper** 服务帐户提交，并确认节点的身份。

- 若要单独批准，请对每个有效的 CSR 运行以下命令：

```
$ oc adm certificate approve <csr_name> 1
```

1 **<csr_name>** 是当前 CSR 列表中 CSR 的名称。

- 要批准所有待处理的 CSR，请运行以下命令：

```
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}{{"\n"}}
{{end}}{{end}}' | xargs --no-run-if-empty oc adm certificate approve
```

注意

在有些 CSR 被批准前，一些 Operator 可能无法使用。

4. 现在，您的客户端请求已被批准，您必须查看添加到集群中的每台机器的服务器请求：

```
$ oc get csr
```

输出示例

```
NAME        AGE    REQUESTOR                                                   CONDITION
csr-bfd72   5m26s  system:node:ip-10-0-50-126.us-east-2.compute.internal
Pending
csr-c57lv   5m26s  system:node:ip-10-0-95-157.us-east-2.compute.internal
Pending
...
```

5. 如果剩余的 CSR 没有被批准，且处于 **Pending** 状态，请批准集群机器的 CSR：

- 若要单独批准，请对每个有效的 CSR 运行以下命令：

```
$ oc adm certificate approve <csr_name> 1
```

1 **<csr_name>** 是当前 CSR 列表中 CSR 的名称。

- 要批准所有待处理的 CSR，请运行以下命令：

```
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}{{"\n"}}
{{end}}{{end}}' | xargs oc adm certificate approve
```

6. 批准所有客户端和服务器 CSR 后，器将处于 **Ready** 状态。运行以下命令验证：

```
$ oc get nodes
```

输出示例

```
NAME      STATUS    ROLES   AGE  VERSION
master-0  Ready     master  73m  v1.20.0
master-1  Ready     master  73m  v1.20.0
master-2  Ready     master  74m  v1.20.0
worker-0  Ready     worker  11m  v1.20.0
worker-1  Ready     worker  11m  v1.20.0
```

注意

批准服务器 CSR 后可能需要几分钟时间让机器转换为 **Ready** 状态。

其他信息

- 如需有关 CSR 的更多信息，请参阅证书签名请求。

8.1.13. 初始 Operator 配置

在 control plane 初始化后，您必须立即配置一些 Operator 以便它们都可用。

先决条件

- 您的 control plane 已初始化。

流程

1. 观察集群组件上线：

```
$ watch -n5 oc get clusteroperators
```

输出示例

```
NAME                                       VERSION AVAILABLE   PROGRESSING   DEGRADED
SINCE
authentication                              4.6.0   True        False         False      3h56m
cloud-credential                            4.6.0   True        False         False      29h
cluster-autoscaler                          4.6.0   True        False         False      29h
config-operator                             4.6.0   True        False         False      6h39m
console                                    4.6.0   True        False         False      3h59m
csi-snapshot-controller                          4.6.0   True        False         False      4h12m
```

```
dns                                        4.6.0   True        False         False      4h15m
etcd                                       4.6.0   True        False         False      29h
image-registry                             4.6.0   True        False         False      3h59m
ingress                                    4.6.0   True        False         False      4h30m
insights                                   4.6.0   True        False         False      29h
kube-apiserver                             4.6.0   True        False         False      29h
kube-controller-manager                    4.6.0   True        False         False      29h
kube-scheduler                             4.6.0   True        False         False      29h
kube-storage-version-migrator              4.6.0   True        False         False      4h2m
machine-api                                4.6.0   True        False         False      29h
machine-approver                           4.6.0   True        False         False      6h34m
machine-config                             4.6.0   True        False         False      3h56m
marketplace                                4.6.0   True        False         False      4h2m
monitoring                                 4.6.0   True        False         False      6h31m
network                                    4.6.0   True        False         False      29h
node-tuning                                4.6.0   True        False         False      4h30m
openshift-apiserver                        4.6.0   True        False         False      3h56m
openshift-controller-manager               4.6.0   True        False         False      4h36m
openshift-samples                          4.6.0   True        False         False      4h30m
operator-lifecycle-manager                 4.6.0   True        False         False      29h
operator-lifecycle-manager-catalog         4.6.0   True        False         False      29h
operator-lifecycle-manager-packageserver   4.6.0   True        False         False      3h59m
service-ca                                 4.6.0   True        False         False      29h
storage                                    4.6.0   True        False         False      4h30m
```

2. 配置不可用的 Operator。

8.1.13.1. 镜像 registry 存储配置

对于不提供默认存储的平台，Image Registry Operator 最初将不可用。安装后，您必须配置 registry 使用的存储，这样 Registry Operator 才可用。

示配置生产集群所需的持久性卷的说明。如果适用，显示有关将空目录配置为存储位置的说明，该位置只可用于非生产集群。

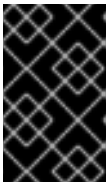
另外还提供了在升级过程中使用 **Recreate** rollout 策略来允许镜像 registry 使用块存储类型的说明。

8.1.13.1.1. 为 IBM Power 系统配置容器镜像仓库（registry）存储

作为集群管理员，在安装后需要配置 registry 来使用存储。

先决条件

- 具有 Cluster Administrator 权限
- IBM Power 系统上的集群。
- 为集群置备的持久性存储，如 Red Hat OpenShift Container Storage。



重要

如果您只有一个副本，OpenShift Container Platform 支持对镜像 registry 存储的 **ReadWriteOnce** 访问。要部署支持高可用性的、带有两个或多个副本的镜像 registry，需要 **ReadWriteMany** 访问设置。

- 必须具有 100Gi 容量。

流程

1. 为了配置 registry 使用存储，需要修改 **configs.imageregistry/cluster** 资源中的 **spec.storage.pvc** 。

注意

使用共享存储时，请查看您的安全设置以防止被外部访问。

2. 验证您没有 registry pod:

```
$ oc get pod -n openshift-image-registry
```

注意

如果存储类型为 **emptyDIR**，则副本数不能超过 **1**。

3. 检查 registry 配置：

```
$ oc edit configs.imageregistry.operator.openshift.io
```

输出示例

```
storage:
  pvc:
    claim:
```

将 **claim** 字段留空以允许自动创建一个 **image-registry-storage** PVC。

4. 检查 **clusteroperator** 的状态：

```
$ oc get clusteroperator image-registry
```

5. 确保您的 registry 设置为 manage，以启用镜像的构建和推送。

- 运行：

```
$ oc edit configs.imageregistry/cluster
```

然后将行改

```
managementState: Removed
```

为

```
managementState: Managed
```

8.1.13.1.2. 在非生产集群中配置镜像 registry 存储

您必须为 Image Registry Operator 配置存储。对于非生产集群，您可以将镜像 registry 设置为空目录。如果您这样做，重启 registry 后会丢失所有镜像。

流程

- 将镜像 registry 存储设置为空目录：

```
$ oc patch configs.imageregistry.operator.openshift.io cluster --type merge --patch '{"spec":{"storage":{"emptyDir":{}}}}'
```

警告

仅可为非生产集群配置这个选项。

如果在 Image Registry Operator 初始化其组件前运行此命令，**oc patch** 命令会失败并显示以下错误：

```
Error from server (NotFound): configs.imageregistry.operator.openshift.io "cluster" not found
```

等待几分钟，然后再次运行该命令。

8.1.14. 在用户置备的基础架构上完成安装

完成 Operator 配置后，可以在您提供的基础架构上完成集群安装。

先决条件

- 您的 control plane 已初始化。
- 已完成初始 Operator 配置。

流程

1. 使用以下命令确认所有集群组件都已在线：

```
$ watch -n5 oc get clusteroperators
```

输出示例

```
NAME                                       VERSION AVAILABLE   PROGRESSING   DEGRADED
SINCE
authentication                             4.6.0   True        False         False     3h56m
cloud-credential                           4.6.0   True        False         False     29h
cluster-autoscaler                         4.6.0   True        False         False     29h
config-operator                            4.6.0   True        False         False     6h39m
console                                    4.6.0   True        False         False     3h59m
csi-snapshot-controller                    4.6.0   True        False         False     4h12m
dns                                        4.6.0   True        False         False     4h15m
```

```
etcd                                       4.6.0   True        False         False      29h
image-registry                             4.6.0   True        False         False      3h59m
ingress                                    4.6.0   True        False         False      4h30m
insights                                   4.6.0   True        False         False      29h
kube-apiserver                             4.6.0   True        False         False      29h
kube-controller-manager                    4.6.0   True        False         False      29h
kube-scheduler                             4.6.0   True        False         False      29h
kube-storage-version-migrator              4.6.0   True        False         False      4h2m
machine-api                                4.6.0   True        False         False      29h
machine-approver                           4.6.0   True        False         False      6h34m
machine-config                             4.6.0   True        False         False      3h56m
marketplace                                4.6.0   True        False         False      4h2m
monitoring                                 4.6.0   True        False         False      6h31m
network                                    4.6.0   True        False         False      29h
node-tuning                                4.6.0   True        False         False      4h30m
openshift-apiserver                        4.6.0   True        False         False      3h56m
openshift-controller-manager               4.6.0   True        False         False      4h36m
openshift-samples                          4.6.0   True        False         False      4h30m
operator-lifecycle-manager                 4.6.0   True        False         False      29h
operator-lifecycle-manager-catalog         4.6.0   True        False         False      29h
operator-lifecycle-manager-packageserver   4.6.0   True        False         False      3h59m
service-ca                                 4.6.0   True        False         False      29h
storage                                    4.6.0   True        False         False      4h30m
```

或者，通过以下命令，如果所有集群都可用您会接到通知。它还检索并显示凭证：

```
$ ./openshift-install --dir <installation_directory> wait-for install-complete 1
```

1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

输出示例

```
INFO Waiting up to 30m0s for the cluster to initialize...
```

Cluster Version Operator 完成从 Kubernetes API 服务器部署 OpenShift Container Platform 集群时，命令运行成功。

重要

- 安装程序生成的 Ignition 配置文件包含在 24 小时后过期的证书，然后在过期时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外情况是，您需要手动批准待处理的 **node-bootstrapper** 证书签名请求（CSR）来恢复 kubelet 证书。如需更多信息，请参阅*从过期的 control plane 证书中恢复*的文档。
- 建议您在生成 12 小时后使用 Ignition 配置文件，因为集群安装后 24 小时证书从 16 小时轮转至 22 小时。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中运行证书更新时避免安装失败。

2. 确认 Kubernetes API 服务器正在与 pod 通信。

   a. 要查看所有 pod 的列表，请使用以下命令：

```
$ oc get pods --all-namespaces
```

输出示例

```
NAMESPACE                    NAME                                        READY   STATUS
RESTARTS   AGE
openshift-apiserver-operator    openshift-apiserver-operator-85cb746d55-zqhs8   1/1
Running    1          9m
openshift-apiserver          apiserver-67b9g                             1/1     Running    0
3m
openshift-apiserver          apiserver-ljcmx                             1/1     Running    0
1m
openshift-apiserver          apiserver-z25h4                             1/1     Running    0
2m
openshift-authentication-operator authentication-operator-69d5d8bf84-vh2n8       1/1
Running    0          5m
...
```

b. 使用以下命令，查看上一命令的输出中所列 pod 的日志：

```
$ oc logs <pod_name> -n <namespace>
```
(1)

(1) 指定 pod 名称和命名空间，如上一命令的输出中所示。

如果 pod 日志显示，Kubernetes API 服务器可以与集群机器通信。

8.1.15. OpenShift Container Platform 的 Telemetry 访问

在 OpenShift Container Platform 4.6 中，默认运行的 Telemetry 服务提供有关集群健康状况和成功更新的指标，需要访问互联网。如果您的集群连接到互联网，Telemetry 会自动运行，而且集群会注册到 OpenShift Cluster Manager。

确认 OpenShift Cluster Manager 清单正确后，可以由 Telemetry 自动维护，也可以使用 OpenShift Cluster Manager 手动维护，使用订阅监控来跟踪帐户或多集群级别的 OpenShift Container Platform 订阅。

其他资源

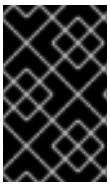
- 有关 Telemetry 服务的更多信息，请参阅关于远程健康监控。

8.1.16. 后续步骤

- 自定义集群。
- 如果需要，您可以选择不使用远程健康报告。

8.2. 在受限网络中的 IBM POWER SYSTEMS 上安装集群

在 OpenShift Container Platform 版本 4.6 中，您可以在受限网络中置备的 IBM Power Systems 基础架构上安装集群。



重要

非裸机平台还有其他注意事项。在尝试在此类环境中安装 OpenShift Container Platform 集群前，请参阅有关在未经测试的平台上部署 OpenShift Container Platform 的指南 中的信息。

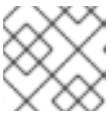
先决条件

- 在受限网络中创建镜像 registry，并获取您的 OpenShift Container Platform 版本的 **imageContentSources** 数据。
- 在开始安装前，必须移动或删除现有的安装文件。这可保证在安装过程中创建和更新所需的安装文件。

重要

确定在可访问安装介质的机器中执行安装步骤。

- 为集群置备持久性存储。若要部署私有镜像 registry，您的存储必须提供 **ReadWriteMany** 访问模式。
- 查看有关 OpenShift Container Platform 安装和更新 流程的详细信息。
- 如果使用防火墙并计划使用遥测（telemetry），您必须将防火墙配置为允许集群需要访问的站点。



注意

如果您要配置代理，请务必也要查看此站点列表。

8.2.1. 关于在受限网络中安装

在 OpenShift Container Platform 4.6 中，可以执行不需要有效的互联网连接来获取软件组件的安装。受限网络安装可使用安装程序置备的基础架构或用户置备的基础架构完成，具体取决于您要安装集群的云平台。

要完成受限网络安装，您必须创建一个 registry，镜像 OpenShift Container Platform registry 的内容并包含其安装介质。您可以在堡垒主机上创建此镜像，该主机可同时访问互联网和您的封闭网络，也可以使用满足您的限制条件的其他方法。

重要

由于用户置备安装配置的复杂性，在尝试使用用户置备的基础架构受限网络安装前，请考虑完成标准用户置备的基础架构安装。通过完成此测试安装，您可以更轻松地隔离和排查您在受限网络中安装时可能出现的问题。

8.2.1.1. 其他限制

受限网络中的集群还有以下额外限制：

- **ClusterVersion** 状态包含一个 **Unable to retrieve available updates** 错误。
- 默认情况下，您无法使用 Developer Catalog 的内容，因为您无法访问所需的镜像流标签。

8.2.2. OpenShift Container Platform 的互联网访问

在 OpenShift Container Platform 4.6 中，您需要访问互联网来获得用来安装集群的镜像。

您必须具有以下互联网访问权限：

- 访问 OpenShift Cluster Manager 以下载安装程序并执行订阅管理。如果集群可以访问互联网，并且没有禁用 Telemetry，该服务会自动授权您的集群。
- 访问 Quay.io，以获取安装集群所需的软件包。
- 获取执行集群更新所需的软件包。

重要

如果您的集群无法直接访问互联网，则可以在置备的某些类基础架构上执行受限网络安装。在此过程中，您要下载所需的内容，并使用它在镜像 registry（mirror registry） 中填充安装集群并生成安装程序所需的软件包。对于某些安装类型，集群要安装到的环境不需要访问互联网。在更新集群之前，要更新 registry 镜像系统中的内容。

8.2.3. 具有用户置备基础架构的集群的机器要求

对于含有用户置备的基础架构的集群，您必须部署所有所需的机器。

8.2.3.1. 所需的机器

最小的 OpenShift Container Platform 集群需要下列主机：

- 一个临时 bootstrap 机器
- 三台 control plane 或 master 机器
- 至少两台计算机器，也称为 worker 机器。

注意

集群要求 bootstrap 机器在三台 control plane 机器上部署 OpenShift Container Platform 集群。您可在安装集群后删除 bootstrap 机器。

重要

要保持集群的高可用性，请将独立的物理主机用于这些集群机器。

bootstrap 和 control plane 机器必须使用 Red Hat Enterprise Linux CoreOS (RHCOS) 作为操作系统。但是，计算机器可以在 Red Hat Enterprise Linux CoreOS(RHCOS)或 Red Hat Enterprise Linux(RHEL)7.9 之间进行选择。

请注意，RHCOS 基于 Red Hat Enterprise Linux（RHEL） 8，并继承其所有硬件认证和要求。请查看Red Hat Enterprise Linux 技术功能及限制 。

8.2.3.2. 网络连接要求

所有 Red Hat Enterprise Linux CoreOS (RHCOS) 机器在启动过程中需要 **initramfs** 中的网络从 Machine Config Server 获取 Ignition 配置文件。在初次启动过程中，需要一个 DHCP 服务器或设置了静态 IP 地址

来建立网络连接，以下载它们的 Ignition 配置文件。另外，集群中的每个 OpenShift Container Platform 节点都必须有权访问网络时间协议（NTP）服务器。如果 DHCP 服务器提供 NTP 服务器信息，Red Hat Enterprise Linux CoreOS（RHCOS）机器上的 chrony 时间服务会读取信息，并可与 NTP 服务器同步时钟。

8.2.3.3. 最低资源要求

每台集群机器都必须满足以下最低要求：

表 8.8. 最低资源要求

机器	操作系统	vCPU [1]	虚拟内存	存储	IOPS [2]
bootstrap	RHCOS	2	16 GB	100 GB	300
Control plane	RHCOS	2	16 GB	100 GB	300
Compute	RHCOS	2	8 GB	100 GB	300

1. 当未启用并发多线程 (SMT) 或超线程时，一个 vCPU 相当于一个物理内核。启用后，使用以下公式来计算对应的比例：（每个内核数的线程）× sockets = vCPU。

2. OpenShift Container Platform 和 Kubernetes 对磁盘性能非常敏感，建议使用更快的存储速度，特别是 control plane 节点上需要 10 ms p99 fsync 持续时间的 etcd。请注意，在许多云平台上，存储大小和 IOPS 可一起扩展，因此您可能需要过度分配存储卷来获取足够的性能。

8.2.3.4. 证书签名请求管理

在使用您置备的基础架构时，集群只能有限地访问自动机器管理，因此您必须提供一种在安装后批准集群证书签名请求 (CSR) 的机制。**kube-controller-manager** 只能批准 kubelet 客户端 CSR。 **machine-approver** 无法保证使用 kubelet 凭证请求的提供证书的有效性，因为它不能确认是正确的机器发出了该请求。您必须决定并实施一种方法，以验证 kubelet 提供证书请求的有效性并进行批准。

8.2.4. 创建用户置备的基础架构

在部署采用用户置备的基础架构的 OpenShift Container Platform 集群前，您必须创建底层基础架构。

先决条件

- 在为集群创建支持基础架构之前，请参阅OpenShift Container Platform 4.x Tested Integrations页。

流程

1. 在每个节点上配置 DHCP 或设置静态 IP 地址。
2. 提供所需的负载均衡器。
3. 配置机器的端口。
4. 配置 DNS。
5. 确保网络可以正常工作。

8.2.4.1. 用户置备的基础架构对网络的要求

所有 Red Hat Enterprise Linux CoreOS（RHCOS）机器在启动过程中需要 **initramfs** 中的网络从机器配置服务器获取 Ignition 配置。

在初次启动过程中，需要一个 DHCP 服务器或集群中的每个机器都设置了静态 IP 地址来建立网络连接，以下载它们的 Ignition 配置文件。

建议您使用 DHCP 服务器为集群进行长期机器管理。确保 DHCP 服务器已配置为向集群机器提供持久 IP 地址和主机名。

Kubernetes API 服务器必须能够解析集群机器的节点名称。如果 API 服务器和 worker 节点位于不同的区域中，您可以配置默认 DNS 搜索区域，以便 API 服务器能够解析节点名称。另一种支持的方法是始终在节点对象和所有 DNS 请求中使用完全限定域名来指代主机。

您必须配置机器间的网络连接，以便集群组件进行通信。每台机器都必须能够解析集群中所有其他机器的主机名。

表 8.9. 所有机器到所有机器

协议	端口	描述
ICMP	N/A	网络可访问性测试
TCP	**1936**	指标
	9000-**9999**	主机级别的服务，包括端口 **9100**-**9101** 上的节点导出器和端口 **9099** 上的 Cluster Version Operator。
	10250-**10259**	Kubernetes 保留的默认端口
	10256	openshift-sdn
UDP	**4789**	VXLAN 和 Geneve
	6081	VXLAN 和 Geneve
	9000-**9999**	主机级别的服务，包括端口 **9100**-**9101** 上的节点导出器。
TCP/UDP	**30000**-**32767**	Kubernetes 节点端口

表 8.10. 要通过控制平面的所有机器

协议	端口	描述
TCP	**6443**	Kubernetes API

表 8.11. control plane 机器到 control plane 机器

协议	端口	描述
TCP	**2379-2380**	etcd 服务器和对等端口

网络拓扑要求
您为集群置备的基础架构必须满足下列网络拓扑要求。

负载均衡器
在安装 OpenShift Container Platform 前，您必须置备两个满足以下要求的负载均衡器：

1. **API 负载均衡器**：提供一个通用端点，供用户（包括人和机器）与平台交互和配置。配置以下条件：

   - 只适用于第 4 层负载均衡。这可被称为 Raw TCP、SSL Passthrough 或者 SSL 桥接模式。如果使用 SSL Bridge 模式，必须为 API 路由启用 Server Name Indication（SNI）。
   - 无状态负载平衡算法。这些选项根据负载均衡器的实现而有所不同。

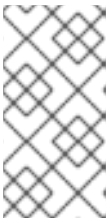
   重要

   不要为 API 负载均衡器配置会话持久性。

   在负载均衡器的前端和后台配置以下端口：

   表 8.12. API 负载均衡器

端口	后端机器（池成员）	内部	外部	描述
6443	Bootstrap 和 control plane.bootstrap 机器初始化集群 control plane 后，您要从负载均衡器中删除 bootstrap 机器。您必须为 API 服务器健康检查探测配置 **/readyz** 端点。	X	X	Kubernetes API 服务器
22623	Bootstrap 和 control plane.bootstrap 机器初始化集群 control plane 后，您要从负载均衡器中删除 bootstrap 机器。	X		机器配置服务器

   注意

   负载均衡器必须配置为，从 API 服务器关闭 **/readyz** 端点到从池中删除 API 服务器实例时最多需要 30 秒。在 **/readyz** 返回错误或处于健康状态后的时间范围内，端点必须被删除或添加。每 5 秒或 10 秒探测一次，有两个成功请求处于健康状态，三个成为不健康的请求经过测试。

2. **应用程序入口负载均衡器**:提供来自集群外部的应用程序流量流量的 Ingress 点。配置以下条件：

   - 只适用于第 4 层负载均衡。这可被称为 Raw TCP、SSL Passthrough 或者 SSL 桥接模式。如果使用 SSL Bridge 模式，您必须为 Ingress 路由启用Server Name Indication（SNI）。

- 建议根据可用选项以及平台上托管的应用程序类型，使用基于连接的或者基于会话的持久性。

在负载均衡器的前端和后台配置以下端口：

表 8.13. 应用程序入口负载均衡器

端口	后端机器（池成员）	内部	外部	描述
443	默认运行入口路由器 Pod、计算或 worker 的机器。	X	X	HTTPS 流量
80	默认运行入口路由器 Pod、计算或 worker 的机器。	X	X	HTTP 流量

提示

如果负载均衡器可以看到客户端的真实 IP 地址，启用基于 IP 的会话持久性可提高使用端到端 TLS 加密的应用程序的性能。

注意

OpenShift Container Platform 集群需要正确配置入口路由器。control plane 初始化后，您必须配置入口路由器。

NTP 配置

OpenShift Container Platform 集群默认配置为使用公共网络时间协议（NTP）服务器。如果要使用本地企业 NTP 服务器，或者集群部署在断开连接的网络中，您可以将集群配置为使用特定的时间服务器。如需更多信息，请参阅*配置 chrony 时间服务*的文档。

如果 DHCP 服务器提供 NTP 服务器信息，Red Hat Enterprise Linux CoreOS（RHCOS）机器上的 chrony 时间服务会读取信息，并可与 NTP 服务器同步时钟。:!restricted:

其他资源

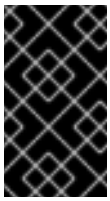
- 配置 chrony 时间服务

8.2.4.2. 用户置备 DNS 要求

DNS 用于名称解析和反向名称解析。DNS A/AAAA 或 CNAME 记录用于名称解析，PTR 记录用于反向解析名称。反向记录很重要，因为 Red Hat Enterprise Linux CoreOS（RHCOS）使用反向记录为所有节点设置主机名。另外，反向记录用于生成 OpenShift Container Platform 需要操作的证书签名请求（CSR）。

采用用户置备的基础架构的 OpenShift Container Platform 集群需要以下 DNS 记录。在每一记录中，**<cluster_name>** 是集群名称，**<base_domain>** 则是您在 **install-config.yaml** 文件中指定的集群基域。完整的 DNS 记录采用如下格式: **<component>.<cluster_name>.<base_domain>.**。

表 8.14. 所需的 DNS 记录

组件	记录	描述
Kubernetes API	**api.<cluster_name>.<base_domain>.**	添加 DNS A/AAAA 或 CNAME 记录，以及 DNS PTR 记录，以识别 control plane 机器的负载均衡器。这些记录必须由集群外的客户端以及集群中的所有节点解析。
	api-int.<cluster_name>.<base_domain>.	添加 DNS A/AAAA 或 CNAME 记录，以及 DNS PTR 记录，以识别 control plane 机器的负载均衡器。这些记录必须可以从集群中的所有节点解析。  **重要** API 服务器必须能够根据在 Kubernetes 中记录的主机名解析 worker 节点。如果 API 服务器无法解析节点名称，则代理的 API 调用会失败，且您无法从 pod 检索日志。
Routes	***.apps.<cluster_name>.<base_domain>.**	添加通配符 DNS A/AAAA 或 CNAME 记录，指向以运行入口路由器 Pod 的机器（默认为 worker 节点）为目标的负载均衡器。这些记录必须由集群外的客户端以及集群中的所有节点解析。
bootstrap	**bootstrap.<cluster_name>.<base_domain>.**	添加 DNS A/AAAA 或 CNAME 记录，以及 DNS PTR 记录来识别 bootstrap 机器。这些记录必须由集群中的节点解析。
Master 主机	**<master><n>.<cluster_name>.<base_domain>.**	DNS A/AAAA 或 CNAME 记录，以识别 control plane 节点（也称为 master 节点）的每台机器。这些记录必须由集群中的节点解析。
Worker 主机	**<worker><n>.<cluster_name>.<base_domain>.**	添加 DNS A/AAAA 或 CNAME 记录，以识别 worker 节点的每台机器。这些记录必须由集群中的节点解析。

提示

您可以使用 **nslookup <hostname>** 命令来验证名称解析。您可以使用 **dig -x <ip_address>** 命令来验证 PTR 记录的反向名称解析。

下面的 BIND 区文件的例子展示了关于名字解析的 A 记录的例子。这个示例的目的是显示所需的记录。这个示例不是为选择一个名称解析服务提供建议。

例 8.3. DNS 区数据库示例

```
$TTL 1W
@ IN SOA ns1.example.com. root (
  2019070700 ; serial
  3H  ; refresh (3 hours)
  30M  ; retry (30 minutes)
  2W  ; expiry (2 weeks)
```

```
   1W )  ; minimum (1 week)
 IN NS ns1.example.com.
 IN MX 10 smtp.example.com.
;
;
ns1 IN A 192.168.1.5
smtp IN A 192.168.1.5
;
helper IN A 192.168.1.5
helper.ocp4 IN A 192.168.1.5
;
; The api identifies the IP of your load balancer.
api.ocp4  IN A 192.168.1.5
api-int.ocp4  IN A 192.168.1.5
;
; The wildcard also identifies the load balancer.
*.apps.ocp4  IN A 192.168.1.5
;
; Create an entry for the bootstrap host.
bootstrap.ocp4 IN A 192.168.1.96
;
; Create entries for the master hosts.
master0.ocp4  IN A 192.168.1.97
master1.ocp4  IN A 192.168.1.98
master2.ocp4  IN A 192.168.1.99
;
; Create entries for the worker hosts.
worker0.ocp4  IN A 192.168.1.11
worker1.ocp4  IN A 192.168.1.7
;
;EOF
```

下面的 BIND 区文件示例显示了反向名字解析的 PTR 记录示例。

例 8.4. 反向记录的 DNS 区数据库示例

```
$TTL 1W
@ IN SOA ns1.example.com. root (
   2019070700 ; serial
   3H  ; refresh (3 hours)
   30M  ; retry (30 minutes)
   2W  ; expiry (2 weeks)
   1W )  ; minimum (1 week)
 IN NS ns1.example.com.
;
; The syntax is "last octet" and the host must have an FQDN
; with a trailing dot.
97 IN PTR master0.ocp4.example.com.
98 IN PTR master1.ocp4.example.com.
99 IN PTR master2.ocp4.example.com.
;
96 IN PTR bootstrap.ocp4.example.com.
;
5 IN PTR api.ocp4.example.com.
```

```
5 IN PTR api-int.ocp4.example.com.
;
11 IN PTR worker0.ocp4.example.com.
7 IN PTR worker1.ocp4.example.com.
;
;EOF
```

8.2.5. 生成 SSH 私钥并将其添加到代理中

如果要在集群上执行安装调试或灾难恢复，则必须为 **ssh-agent** 和安装程序提供 SSH 密钥。您可以使用此密钥访问公共集群中的 bootstrap 机器来排除安装问题。

注意

在生产环境中，您需要进行灾难恢复和调试。

您可以使用此密钥以 **core** 用户身份通过 SSH 连接到 master 节点。在部署集群时，此密钥会添加到 **core** 用户的 **~/.ssh/authorized_keys** 列表中。

注意

您必须使用一个本地密钥，而不要使用在特定平台上配置的密钥，如 AWS 密钥对。

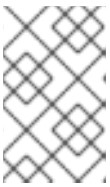
流程

1. 如果还没有为计算机上免密码身份验证而配置的 SSH 密钥，请创建一个。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ ssh-keygen -t ed25519 -N '' \
  -f <path>/<file_name> 1
```

1 指定新 SSH 密钥的路径和文件名，如 **~/.ssh/id_rsa**。如果您已有密钥对，请确保您的公钥位于 **~/.ssh** 目录中。

运行此命令会在指定的位置生成不需要密码的 SSH 密钥。



注意

如果您计划在 **x86_64** 架构中安装使用 FIPS 验证的/Modules in Process 加密库的 OpenShift Container Platform 集群，不要创建使用 **ed25519** 算法的密钥。反之，创建一个使用 **rsa** 或 **ecdsa** 算法的密钥。

2. 作为后台任务启动 **ssh-agent** 进程：

```
$ eval "$(ssh-agent -s)"
```

输出示例

```
Agent pid 31874
```

注意

如果您的集群采用 FIPS 模式，则只使用 FIPS 兼容算法来生成 SSH 密钥。密钥必须是 RSA 或 ECDSA。

3. 将 SSH 私钥添加到 **ssh-agent**：

```
$ ssh-add <path>/<file_name> 1
```

输出示例

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

1 指定 SSH 私钥的路径和文件名，如 **~/.ssh/id_rsa**

后续步骤

- 在安装 OpenShift Container Platform 时，为安装程序提供 SSH 公钥。

8.2.6. 手动创建安装配置文件

对于使用用户置备的基础架构的 OpenShift Container Platform 安装，您必须手动生成安装配置文件。

先决条件

- 获取 OpenShift Container Platform 安装程序和集群的访问令牌。
- 获取命令输出中的 **imageContentSources** 部分来镜像存储库。
- 获取您的镜像 registry 的证书内容。

流程

1. 创建用来存储您所需的安装资产的安装目录：

```
$ mkdir <installation_directory>
```

重要

您必须创建目录。一些安装信息，如 bootstrap X.509 证书，有较短的过期间隔，因此不要重复使用安装目录。如果要重复使用另一个集群安装中的个别文件，可以将其复制到您的目录中。但是，一些安装数据的文件名可能会在发行版本之间有所改变。从 OpenShift Container Platform 老版本中复制安装文件时要格外小心。

2. 自定义以下 **install-config.yaml** 文件模板，并将它保存到 **<installation_directory>** 中。

注意

此配置文件必须命名为 **install-config.yaml**。

- 除非使用 RHCOS 默认信任的 registry，如 **docker.io**，否则必须在 **additionalTrustBundle** 部分中提供镜像存储库的证书内容。在大多数情况下，必须为您的镜像提供证书。
- 您必须包含命令输出中的 **imageContentSources** 部分，才能镜像存储库。

3. 备份 **install-config.yaml** 文件，以便用于安装多个集群。

重要

install-config.yaml 文件会在安装过程的下一步骤中消耗掉。现在必须备份它。

8.2.6.1. IBM Power 系统的 install-config.yaml 文件示例

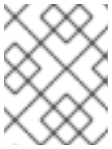
您可以自定义 **install-config.yaml** 文件，以指定有关 OpenShift Container Platform 集群平台的更多信息，或修改所需参数的值。

```
apiVersion: v1
baseDomain: example.com (1)
compute: (2)
- hyperthreading: Enabled (3)
  name: worker
  replicas: 0 (4)
  architecture : ppc64le
controlPlane: (5)
  hyperthreading: Enabled (6)
  name: master
  replicas: 3 (7)
  architecture : ppc64le
metadata:
  name: test (8)
networking:
  clusterNetwork:
  - cidr: 10.128.0.0/14 (9)
    hostPrefix: 23 (10)
  networkType: OpenShiftSDN
  serviceNetwork: (11)
  - 172.30.0.0/16
platform:
  none: {} (12)
fips: false (13)
pullSecret: '{"auths":{"<local_registry>": {"auth": "<credentials>","email": "you@example.com"}}}' (14)
sshKey: 'ssh-ed25519 AAAA...' (15)
additionalTrustBundle: | (16)
  -----BEGIN CERTIFICATE-----
  ZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZ
  -----END CERTIFICATE-----
imageContentSources: (17)
- mirrors:
  - <local_registry>/<local_repository_name>/release
  source: quay.io/openshift-release-dev/ocp-release
- mirrors:
  - <local_registry>/<local_repository_name>/release
  source: quay.io/openshift-release-dev/ocp-v4.0-art-dev
```

1 集群的基域。所有 DNS 记录都必须是这个基域的子域，并包含集群名称。

2 5 **controlPlane** 部分是一个单映射，但 **compute** 部分是一系列映射。为满足不同数据结构的要求，**compute** 部分的第一行必须以连字符 **-** 开头，**controlPlane** 部分的第一行则不可以连字符开头。只使用一个 control plane 池。

3 6 是否要启用或禁用并发多线程（SMT）或**超线程**。默认情况下，启用 SMT 可提高机器内核的性能。您可以通过将参数值设为 **Disabled** 来禁用。如果禁用 SMT，则必须在所有集群机器中禁用它，其中包括 control plane 和计算机器。



注意

默认启用并发多线程（SMT）。如果在 BIOS 设置中没有启用 SMT，**hyperthreading** 参数不会起作用。

重要

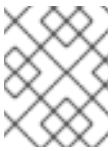
如果您禁用 **hyperthreading**（无论是在 BIOS 中还是在 **install-config.yaml** 中），请确保您对可能会造成的机器性能显著降低的情况有所考虑。

4 **replicas** 参数的值必须设置为 **0**。此参数控制集群为您创建和管理的 worker 数量，使用用户置备的基础架构时集群不会执行这些功能。在完成 OpenShift Container Platform 安装前，您必须手动为集群部署 worker 机器。

7 您添加到集群的 control plane 机器数量。由于集群将这个值用作集群中 etcd 端点的数量，因此该值必须与您部署的 control plane 机器数量匹配。

8 您在 DNS 记录中指定的集群名称。

9 从中分配 pod IP 地址的 IP 地址块。此块不得与现有的物理网络重叠。这些 IP 地址用于 pod 网络。如果您需要从外部网络访问 pod，请配置负载均衡器和路由器来管理流量。



注意

类 E CIDR 范围保留给以后使用。要使用 Class E CIDR 范围，您必须确保您的网络环境接受 Class E CIDR 范围内的 IP 地址。

10 分配给每个单独节点的子网前缀长度。例如，如果 **hostPrefix** 设为 **23**，则每个节点从所给的 **cidr** 中分配一个 **/23** 子网，这样就能有 510 (2^(32 - 23) - 2) 个 Pod IP 地址。如果您需要从外部网络访问节点，请配置负载均衡器和路由器来管理流量。

11 用于服务 IP 地址的 IP 地址池。您只能输入一个 IP 地址池。此块不得与现有的物理网络重叠。如果您需要从外部网络访问服务，请配置负载均衡器和路由器来管理流量。

12 您必须将平台设置为 **none**。您无法为 IBM Power 系统基础架构提供额外的平台配置变量。

13 是否启用或禁用 FIPS 模式。默认情况下不启用 FIPS 模式。如果启用了 FIPS 模式，运行 OpenShift Container Platform 的 Red Hat Enterprise Linux CoreOS (RHCOS) 机器会绕过默认的 Kubernetes 加密套件，并使用由 RHCOS 提供的加密模块。

重要

只有在 **x86_64** 架构中的 OpenShift Container Platform 部署支持 FIPS 验证的/Modules in Process 加密库。

14 对于 **<local_registry>**，请指定 registry 域名，以及您的镜像 registry 用来提供内容的可选端口。例如： **registry.example.com** 或者 **registry.example.com:5000**。使用 **<credentials>** 为您生成的镜

15 Red Hat Enterprise Linux CoreOS (RHCOS) 中 **core** 用户的默认 SSH 密钥的公钥部分。

注意

对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 **ssh-agent** 进程使用的 SSH 密钥。

16 提供用于镜像 registry 的证书文件内容。

17 提供命令输出中的 **imageContentSources** 部分来镜像存储库。

8.2.6.2. 在安装过程中配置集群范围代理

生产环境可能会拒绝直接访问互联网，而是提供 HTTP 或 HTTPS 代理。您可以通过在 **install-config.yaml** 文件中配置代理设置，将新的 OpenShift Container Platform 集群配置为使用代理。

先决条件

- 您有一个现有的 **install-config.yaml** 文件。
- 您检查了集群需要访问的站点，并决定是否需要绕过代理。默认情况下代理所有集群出口流量，包括对托管云供应商 API 的调用。您需要将站点添加到 **Proxy** 对象的 **spec.noProxy** 字段来绕过代理。

注意

Proxy 对象 **status.noProxy** 字段使用安装配置中的 **networking.machineNetwork[].cidr**、**networking.clusterNetwork[].cidr** 和 **networking.serviceNetwork[]** 字段的值填充。

对于在 Amazon Web Services(AWS)、Google Cloud Platform(GCP)、Microsoft Azure 和 Red Hat OpenStack Platform(RHOSP)上安装, **Proxy** 对象 **status.noProxy** 字段也会使用实例元数据端点填充(**169.254.169.254**)。

流程

1. 编辑 **install-config.yaml** 文件并添加代理设置。例如：

```
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: example.com 3
additionalTrustBundle: | 4
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
...
```

1. 用于创建集群外 HTTP 连接的代理 URL。URL 必须是 **http**。
2. 用于创建集群外 HTTPS 连接的代理 URL。
3. 要排除在代理中的目标域名、IP 地址或其他网络 CIDR 的逗号分隔列表。在域前面加 **.** 来仅匹配子域。例如： **.y.com** 匹配 **x.y.com**，但不匹配 **y.com**。使用 * 绕过所有目的地的代理。
4. 如果提供，安装程序会在 **openshift-config 命名空间中生成名为 user-ca- bundle** 的配置映射来保存额外的 CA 证书。如果您提供 **additionalTrustBundle** 和至少一个代理设置，则 **Proxy** 对象会被配置为引用 **trustedCA** 字段中的 **user-ca-bundle** 配置映射。然后，Cluster Network Operator 会创建一个 **trusted-ca-bundle** 配置映射，该配置映射将为 **trustedCA** 参数指定的内容与 RHCOS 信任捆绑包合并。 **additionalTrustBundle** 字段是必需的，除非代理的身份证书由来自 RHCOS 信任捆绑包的颁发机构签名。

注意

安装程序不支持代理的 **readinessEndpoints** 字段。

2. 保存该文件，并在安装 OpenShift Container Platform 时引用。

安装程序会创建一个名为 **cluster** 的集群范围代理，该代理使用提供的 **install-config.yaml** 文件中的代理设置。如果没有提供代理设置，仍然会创建一个 **cluster Proxy** 对象，但它会有一个空 **spec**。

注意

只支持名为 **cluster** 的 **Proxy** 对象，且无法创建额外的代理。

8.2.7. 创建 Kubernetes 清单和 Ignition 配置文件

由于您必须修改一些集群定义文件并要手动启动集群机器，因此您必须生成 Kubernetes 清单和 Ignition 配置文件，集群需要这两项来创建其机器。

安装配置文件转换为 Kubernetes 清单。清单嵌套到 Ignition 配置文件中，稍后用于创建集群。

重要

- 安装程序生成的 Ignition 配置文件包含在 24 小时后过期的证书，然后在过期时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外情况是，您需要手动批准待处理的 **node-bootstrapper** 证书签名请求（CSR）来恢复 kubelet 证书。如需更多信息，请参阅 *从过期的 control plane 证书中恢复*的文档。
- 建议您在生成 12 小时后使用 Ignition 配置文件，因为集群安装后 24 小时证书从 16 小时轮转至 22 小时。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中运行证书更新时避免安装失败。

先决条件

- 已获得 OpenShift Container Platform 安装程序。对于受限网络安装，这些文件位于您的堡垒主机上。
- 已创建 **install-config.yaml** 安装配置文件。

流程

1. 切换到包含安装程序的目录，并为集群生成 Kubernetes 清单：

```
$ ./openshift-install create manifests --dir <installation_directory> 1
```

1 对于 **<installation_directory>**，请指定含有您创建的 **install-config.yaml** 文件的安装目录。

2. 检查 **<installation_directory>/manifests/cluster-scheduler-02-config.yml** Kubernetes 清单文件中的 **mastersSchedulable** 参数是否已设置为 **false**。此设置可防止在 control plane 机器上调度 pod:
   a. 打开 **<installation_directory>/manifests/cluster-scheduler-02-config.yml** 文件。
   b. 找到 **mastersSchedulable** 参数并确保它被设置为 **false**。
   c. 保存并退出文件。
3. 要创建 Ignition 配置文件，从包含安装程序的目录运行以下命令：

```
$ ./openshift-install create ignition-configs --dir <installation_directory> 1
```

1 对于 **<installation_directory>**，请指定相同的安装目录。

该目录中将生成以下文件：

```
.
├── auth
│   ├── kubeadmin-password
│   └── kubeconfig
├── bootstrap.ign
├── master.ign
├── metadata.json
└── worker.ign
```

8.2.8. 创建 Red Hat Enterprise Linux CoreOS (RHCOS) 机器

在您置备的 IBM Power Systems 基础架构上安装集群前，必须先创建 RHCOS 机器供它使用。按照相应的步骤，使用 ISO 镜像或网络 PXE 启动来创建机器。

8.2.8.1. 使用 ISO 镜像创建 Red Hat Enterprise Linux CoreOS (RHCOS) 机器

在您置备的 IBM Power Systems 基础架构上安装集群前，必须先创建 RHCOS 机器供它使用。您可以使用 ISO 镜像来创建这些机器。

先决条件

- 获取集群的 Ignition 配置文件。
- 具有可从计算机以及您创建的机器访问的 HTTP 服务器的访问权限。

流程

1. 将安装程序创建的 control plane、计算和 bootstrap Ignition 配置文件上传到 HTTP 服务器。记下这些文件的 URL。

重要

如果您计划在安装完成后在集群中添加更多计算机器，请不要删除这些文件。

2. 从 RHCOS 镜像镜像页面获取您选择的操作系统实例安装方法所需的 RHCOS 镜像。

重要

RHCOS 镜像可能不会随着 OpenShift Container Platform 的每一发行版本都有改变。您必须下载最高版本的镜像，其版本号应小于或等于您安装的 OpenShift Container Platform 版本。如果可用，请使用与 OpenShift Container Platform 版本匹配的镜像版本。此流程只使用 ISO 镜像。此安装类型不支持 RHCOS qcow2 镜像。

ISO 文件名类似以下示例：

rhcos-<version>-live.<architecture>.iso

3. 使用 ISO 启动 RHCOS 安装。使用如下安装选项之一：

- 将 ISO 镜像刻录到磁盘并直接启动。
- 通过 LOM 接口使用 ISO 重定向。

4. 引导 ISO 镜像。您可以中断安装引导过程来添加内核参数。然而，在这个 ISO 过程中，您应该使用 **coreos-installer** 命令而不是添加内核参数。如果您在没有选项或中断的情况下运行 live 安装程序，安装程序将引导至 live 系统上的 shell 提示符，准备好将 RHCOS 安装到磁盘中。

5. 在运行 **coreos-installer** 前，请参阅 *高级 RHCOS 安装参考* 部分，以了解配置功能的不同方法，如网络和磁盘分区。

6. 运行 **coreos-installer** 命令。您至少必须识别节点类型的 Ignition 配置文件位置，以及您要安装到的磁盘位置。下面是一个示例：

```
$ sudo coreos-installer install \
    --ignition-url=https://host/worker.ign /dev/sda
```

7. 安装 RHCOS 后，系统会重启。系统重启过程中，它会应用您指定的 Ignition 配置文件。

8. 继续为集群创建其他机器。

重要

此刻您必须创建 bootstrap 和 control plane 机器。如果 control plane 机器不可调度（这是默认调度），则在安装集群前至少会创建两台计算机器。

8.2.8.1.1. 高级 RHCOS 安装参考

本节演示了网络配置和其他高级选项，允许您修改 Red Hat Enterprise Linux CoreOS（RHCOS）手动安装过程。下表描述了您可以与 RHCOS live installer 和 **coreos-installer** 命令一起使用的内核参数和命令行选项。

RHCOS 启动提示下的路由和绑定选项
如果从 ISO 镜像安装 RHCOS，您可以在引导该镜像时手动添加内核参数以配置节点的网络。如果没有使用网络参数，则安装默认为使用 DHCP。

重要

添加网络参数时，还必须添加 **rd.neednet=1** 内核参数。

下表描述了如何为实时 ISO 安装使用 **ip=** 、 **nameserver=** 和 **bond=** 内核参数。

注意

在添加内核参数时顺序非常重要： **ip=**，**nameserver=**，然后 **bond=**。

ISO 的路由和绑定选项

下表提供了配置 Red Hat Enterprise Linux CoreOS（RHCOS）节点网络的示例。这些是在系统引导过程中传递给 **dracut** 工具的网络选项。有关 **dracut** 支持的网络选项的详情，请参考 **dracut.cmdline** 手册页。

描述	例子
要配置一个 IP 地址，可以使用 DHCP(**ip=dhcp**)或者设置单独的静态 IP 地址(**ip=<host_ip>**)。然后在每个节点上指定 DNS 服务器 IP 地址(**nameserver=<dns_ip>**)。这个示例设置： • 节点的 IP 地址为 **10.10.10.2** • 网关地址为 **10.10.10.254** • 子网掩码为 **255.255.255.0** • 主机名为 **core0.example.com** • DNS 服务器地址为 **4.4.4.41**	ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp1s0:none nameserver=4.4.4.41
通过指定多个 **ip=** 条目来指定多个网络接口。	ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp1s0:none ip=10.10.10.3::10.10.10.254:255.255.255.0:core0.example.com:enp2s0:none

描述	例子

可选：您可以通过设置一个 **rd.route=** 值来配置到额外网络的路由。
 	
如果额外网络网关与主要网络网关不同，则默认网关必须是主要网络网关。
	配置默认网关：
	
	`ip=::10.10.10.254::::`
	
	为额外网络配置路由：
	
	`rd.route=20.20.20.0/24:20.20.20.254:enp2s0`

在单一接口中禁用 DHCP，比如当有两个或者多个网络接口时，且只有一个接口被使用。	`ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp1s0:none` `ip=::::core0.example.com:enp2s0:none`
您可以将系统中 DHCP 和静态 IP 配置与多个网络接口结合在一起。	`ip=enp1s0:dhcp` `ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp2s0:none`

可选： 您可以使用 **vlan=** 参数在单独的接口上配置 VLAN。	在网络接口中配置 VLAN 并使用静态 IP 地址：
	
	`ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp2s0.100:none`
	`vlan=enp2s0.100:enp2s0`
	
	在网络接口中配置 VLAN 并使用 DHCP：
	
	`ip=enp2s0.100:dhcp`
	`vlan=enp2s0.100:enp2s0`

您可以为每个服务器添加一个 **nameserver=** 条目来提供多个 DNS 服务器。	`nameserver=1.1.1.1` `nameserver=8.8.8.8`

可选：使用 **bond=** 选项支持将多个网络接口绑定到一个接口。在这两个示例中：
 	
● 配置绑定接口的语法为：**bond=name[:network_interfaces][:options]**
 	
● *name* 是绑定设备名称（**bond0**），*network_interfaces* 代表用逗号分开的物理（以太网）接口(**em1,em2**)的列表，*options* 是用逗号分开的绑定选项列表。输入 **modinfo bonding** 查看可用选项。
 	
● 当使用 **bond=** 创建绑定接口时，您必须指定如何分配 IP 地址以及绑定接口的其他信息。
	要将绑定的接口配置为使用 DHCP，请将绑定的 IP 地址设置为 **dhcp**。例如：
	
	`bond=bond0:em1,em2:mode=active-backup`
	`ip=bond0:dhcp`
	
	要将绑定接口配置为使用静态 IP 地址，请输入您需要的特定 IP 地址以及相关信息。例如：
	
	`bond=bond0:em1,em2:mode=active-backup`
	`ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:bond0:none`

描述	例子
可选： 您可以使用 **vlan=** 参数在绑定接口上配置 VLAN。	使用 VLAN 配置绑定接口并使用 DHCP： ip=bond0.100:dhcp bond=bond0:em1,em2:mode=active-backup vlan=bond0.100:bond0 使用 VLAN 配置绑定接口，并使用静态 IP 地址： ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:bond0.100:none bond=bond0:em1,em2:mode=active-backup vlan=bond0.100:bond0
可选：通过使用 **team=** 参数，网络合作可用作绑定的替代选择。在此例中： ● 配置组接口的语法为：**team=name[:network_interfaces]** *name* 是组设备名称(**team0**)，**network** *_interfaces* 代表以逗号分隔的物理（以太网）接口（**em1**、**em2**）列表。 **注意** 当 RHCOS 切换到即将推出的 RHEL 版本时，团队计划会被弃用。如需更多信息，请参阅红帽知识库文章。	配置网络团队： team=team0:em1,em2 ip=team0:dhcp

8.2.8.2. 通过 PXE 或 iPXE 启动来创建 Red Hat Enterprise Linux CoreOS (RHCOS) 机器

在安装使用手动置备 RHCOS 节点（如裸机）的集群前，您必须创建 RHCOS 机器供其使用。您可以使用 PXE 或 iPXE 启动来创建机器。

先决条件

- 获取集群的 Ignition 配置文件。
- 配置合适的 PXE 或 iPXE 基础架构。
- 具有 HTTP 服务器的访问权限，以便您可从计算机进行访问。

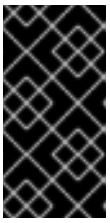
流程

1. 将安装程序创建的 master、worker 和 bootstrap Ignition 配置文件上传到 HTTP 服务器。记下这些文件的 URL。

重要

您可以在 Ignition 配置中添加或更改配置设置，然后将其保存到 HTTP 服务器。如果您计划在安装完成后在集群中添加更多计算机器，请不要删除这些文件。

2. 从 RHCOS 镜像 镜像页面获取 RHCOS **内核、initram fs 和 rootfs** 文件。



重要

RHCOS 工件（artifact）可能不会随着 OpenShift Container Platform 的每个发行版本而改变。您必须下载最高版本的工件，其版本号应小于或等于您安装的 OpenShift Container Platform 版本。这个过程只使用下面描述的正确 **kernel**、**initramfs** 和 **rootfs** 工件。此安装类型不支持 RHCOS qcow2 镜像。

文件名包含 OpenShift Container Platform 版本号。它们类似以下示例：

- **kernel**: **rhcos-<version>-live-kernel-<architecture>**
- **initramfs**: **rhcos-<version>-live-initramfs.<architecture>.img**
- **rootfs**: **rhcos-<version>-live-rootfs.<architecture>.img**

3. 上传引导方法所需的额外文件：
   - 对于传统的 PXE，将 **kernel** 和 **initramfs** 文件上传到 TFTP 服务器，并将 **rootfs** 文件上传到 HTTP 服务器。
   - 对于 iPXE，将 **kernel**、**initram** fs 和 **rootfs** 文件上传到 HTTP 服务器。

重要

如果您计划在安装完成后在集群中添加更多计算机器，请不要删除这些文件。

4. 配置网络启动基础架构，以便在安装 RHCOS 后机器可从本地磁盘启动。

5. 为 RHCOS 镜像配置 PXE 或 iPXE 安装。
针对您的环境修改以下示例菜单条目之一，并验证能否正确访问镜像和 Ignition 文件：
   - 对于 PXE：

```
DEFAULT pxeboot
TIMEOUT 20
PROMPT 0
LABEL pxeboot
    KERNEL http://<HTTP_server>/rhcos-<version>-live-kernel-<architecture> (1)
    APPEND initrd=http://<HTTP_server>/rhcos-<version>-live-initramfs.
<architecture>.img coreos.live.rootfs_url=http://<HTTP_server>/rhcos-<version>-live-
rootfs.<architecture>.img coreos.inst.install_dev=/dev/sda
coreos.inst.ignition_url=http://<HTTP_server>/bootstrap.ign (2) (3)
```

(1) 指定上传到 HTTP 服务器的 live **kernel** 文件位置。URL 必须是 HTTP、TFTP 或者 FTP；不支持 HTTPS 和 NFS。

(2) 如果您使用多个 NIC，请在 **ip** 选项中指定一个接口。例如，要在名为 **eno1** 的 NIC 上使用 DHCP，请设置 **ip=eno1:dhcp**。

(3) 指定上传到 HTTP 服务器的 RHCOS 文件的位置。**initrd** 参数值是 **initramfs** 文件的位置，**coreos.live.rootfs_url** 参数值是 **rootfs** 文件的位置，**coreos.inst.ignition_url** 参数值是 bootstrap Ignition 配置文件的位置。您还可以在 **APPEND** 行中添加更多内核参数来配置联网或其他引导选项。

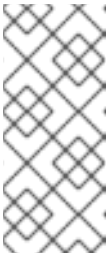
注意

这个配置不会在使用图形控制台的机器上启用串口控制台访问。要配置不同的控制台，请在 **APPEND** 行中添加一个或多个 **console=** 参数。例如，添加 **console=tty0 console=ttyS0** 将第一个 PC 串口设置为主控制台，图形控制台作为二级控制台。如需更多信息，请参阅如何在 Red Hat Enterprise Linux 中设置串行终端和（或）控制台？

- 对于 iPXE：

```
kernel http://<HTTP_server>/rhcos-<version>-live-kernel-<architecture> initrd=main
coreos.live.rootfs_url=http://<HTTP_server>/rhcos-<version>-live-rootfs.
<architecture>.img coreos.inst.install_dev=/dev/sda
coreos.inst.ignition_url=http://<HTTP_server>/bootstrap.ign 1 2
initrd --name main http://<HTTP_server>/rhcos-<version>-live-initramfs.
<architecture>.img 3
boot
```

1. 指定上传到 HTTP 服务器的 RHCOS 文件的位置。**kernel** 参数值是 **kernel** 文件的位置，在 UEFI 系统中引导时需要 **initrd=main** 参数。 **coreos.live.rootfs_url** 参数值是 **rootfs** 文件的位置， **coreos.inst.ignition_url** 参数值则是 bootstrap Ignition 配置文件的位置。
2. 如果您使用多个 NIC，请在 **ip** 选项中指定一个接口。例如，要在名为 **eno1** 的 NIC 上使用 DHCP，请设置 **ip=eno1:dhcp**。
3. 指定上传到 HTTP 服务器的 **initramfs** 文件的位置。



注意

这个配置不会在使用图形控制台的机器上启用串口控制台访问。要配置不同的控制台，请在 **kerne** 行中添加一个或多个 **console=** 参数。例如，添加 **console=tty0 console=ttyS0** 将第一个 PC 串口设置为主控制台，图形控制台作为二级控制台。如需更多信息，请参阅如何在 Red Hat Enterprise Linux 中设置串行终端和（或）控制台？

6. 如果使用 PXE UEFI，请执行以下操作：
   a. 提供引导系统所需的 **shim x64.efi 和 grubx64 .efi EFI 二进制文件以及 grub.cfg** 文件。
      - 通过将 RHCOS ISO 挂载到主机，然后将 **images/efiboot.img** 文件挂载到您的主机来提取所需的 EFI 二进制文件：

```
$ mkdir -p /mnt/iso
```

```
$ mkdir -p /mnt/efiboot
```

```
$ mount -o loop rhcos-installer.x86_64.iso /mnt/iso
```

```
$ mount -o loop,ro /mnt/iso/images/efiboot.img /mnt/efiboot
```

- 从 **efiboot.img** 挂载点，将 **EFI/redhat/shimx64.efi** 和 **EFI/redhat/grubx64.efi** 文件复制到 TFTP 服务器中：

```
$ cp /mnt/efiboot/EFI/redhat/shimx64.efi .
```

```
$ cp /mnt/efiboot/EFI/redhat/grubx64.efi .
```

```
$ umount /mnt/efiboot
```

```
$ umount /mnt/iso
```

- 将 RHCOS ISO 中包含的 **EFI/redhat/grub.cfg** 文件复制到您的 TFTP 服务器中。

b. 编辑 **grub.cfg** 文件使其包含类似如下的参数：

```
menuentry 'Install Red Hat Enterprise Linux CoreOS' --class fedora --class gnu-linux --
class gnu --class os {
 linuxefi rhcos-<version>-live-kernel-<architecture> coreos.inst.install_dev=/dev/sda
coreos.live.rootfs_url=http://<HTTP_server>/rhcos-<version>-live-rootfs.
<architecture>.img coreos.inst.ignition_url=http://<HTTP_server>/bootstrap.ign
 initrdefi rhcos-<version>-live-initramfs.<architecture>.img
}
```

其中：

rhcos-<version>-live-kernel-<architecture>

指定上传到 TFTP 服务器的 **内核** 文件。

http://<HTTP_server>/rhcos-<version>-live-rootfs.<architecture>.img

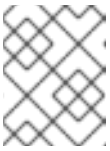
指定上传到 HTTP 服务器的 live rootfs 镜像的位置。

http://<HTTP_server>/bootstrap.ign

指定上传到 HTTP 服务器的 bootstrap Ignition 配置文件的位置。

rhcos-<version>-live-initramfs.<architecture>.img

指定上传到 TFTP 服务器的 **initramfs** 文件的位置。



注意

有关如何为 UEFI 引导配置 PXE 服务器的更多信息，请参阅红帽知识库文章：如何为 Red Hat Enterprise Linux 的 UEFI 引导配置/设置 PXE 服务器？

7. 继续为集群创建机器。

重要

此刻您必须创建 bootstrap 和 control plane 机器。如果 control plane 机器不可调度（这是默认调度），则在安装集群前至少会创建两台计算机器。

8.2.9. 创建集群

要创建 OpenShift Container Platform 集群，请等待您通过安装程序生成的 Ignition 配置文件所置备的机器上完成 bootstrap 过程。

先决条件

- 为集群创建所需的基础架构。
- 已获得安装程序并为集群生成了 Ignition 配置文件。
- 已使用 Ignition 配置文件为集群创建 RHCOS 机器。

流程

1. 监控 bootstrap 过程：

```
$ ./openshift-install --dir <installation_directory> wait-for bootstrap-complete \ 1
    --log-level=info 2
```

1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

2 要查看不同的安装详情，请指定 **warn**、**debug** 或 **error**，而不要指定 **info**。

输出示例

```
INFO Waiting up to 30m0s for the Kubernetes API at https://api.test.example.com:6443...
INFO API v1.19.0 up
INFO Waiting up to 30m0s for bootstrapping to complete...
INFO It is now safe to remove the bootstrap resources
```

Kubernetes API 服务器提示已在 control plane 机器上完成 bootstrap 时，命令运行成功。

2. bootstrap 过程完成后，请从负载均衡器中删除 bootstrap 机器。

重要

此时您必须从负载均衡器中删除 bootstrap 机器。您还可以删除或重新格式化机器本身。

8.2.10. 使用 CLI 登录到集群

您可以通过导出集群 **kubeconfig** 文件，以默认系统用户身份登录集群。**kubeconfig** 文件包含关于集群的信息，供 CLI 用于将客户端连接到正确集群和 API 服务器。该文件特只适用于一个特定的集群，在 OpenShift Container Platform 安装过程中创建。

先决条件

- 已部署了 OpenShift Container Platform 集群。
- 已安装 **oc** CLI。

流程

1. 导出 **kubeadmin** 凭证：

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

2. 使用导出的配置，验证能否成功运行 **oc** 命令：

```
$ oc whoami
```

输出示例

```
system:admin
```

8.2.11. 批准机器的证书签名请求

将机器添加到集群时，会为您添加的每台机器生成两个待处理证书签名请求（CSR）。您必须确认这些 CSR 已获得批准，或根据需要自行批准。客户端请求必须首先被批准，然后是服务器请求。

先决条件

- 您已将机器添加到集群中。

流程

1. 确认集群可以识别这些机器：

```
$ oc get nodes
```

输出示例

```
NAME     STATUS   ROLES   AGE  VERSION
master-0 Ready    master  63m  v1.19.0
master-1 Ready    master  63m  v1.19.0
master-2 Ready    master  64m  v1.19.0
```

输出将列出您创建的所有机器。

注意

在一些 CSR 被批准前，以上输出可能不包括计算节点（也称为 worker 节点）。

2. 检查待处理的 CSR，并确保可以看到添加到集群中的每台机器都有 **Pending** 或 **Approved** 状态的客户端请求：

```
$ oc get csr
```

输出示例

```
NAME        AGE    REQUESTOR                                                   CONDITION
csr-8b2br   15m    system:serviceaccount:openshift-machine-config-operator:node-
bootstrapper   Pending
csr-8vnps   15m    system:serviceaccount:openshift-machine-config-operator:node-
bootstrapper   Pending
...
```

在本例中，两台机器加入了集群。您可能会在列表中看到更多已批准的 CSR。

3. 如果 CSR 没有获得批准，请在所添加机器的所有待处理 CSR 都处于 **Pending** 状态后，为您的集群机器批准这些 CSR：

注意

由于 CSR 会自动轮转，因此请在将机器添加到集群后一小时内批准您的 CSR。如果没有在一小时内批准，证书将会轮转，每个节点将会存在多个证书。您必须批准所有这些证书。批准客户端 CSR 后， Kubelet 为服务证书创建辅助 CSR，这需要手动批准。然后，如果 Kubelet 请求具有相同参数的新证书，则 **machine-approver** 会自动批准后续服务证书续订请求。

注意

对于在未启用机器 API 的平台中运行的集群，如裸机和其他用户置备的基础架构，必须采用一种方法自动批准 kubelet 提供证书请求（CSR）。如果没有批准请求，则 **oc exec**、**oc rsh** 和 **oc logs** 命令将无法成功，因为 API 服务器连接到 kubelet 时需要服务证书。与 Kubelet 端点联系的任何操作都需要此证书批准。这个方法必须监视新的 CSR，确认 CSR 由 **system:node** 或 **system:admin** 组中的 **node-bootstrapper** 服务帐户提交，并确认节点的身份。

- 若要单独批准，请对每个有效的 CSR 运行以下命令：

```
$ oc adm certificate approve <csr_name> 1
```

1 **<csr_name>** 是当前 CSR 列表中 CSR 的名称。

- 要批准所有待处理的 CSR，请运行以下命令：

```
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}{{"\n"}}
{{end}}{{end}}' | xargs --no-run-if-empty oc adm certificate approve
```

注意

在有些 CSR 被批准前，一些 Operator 可能无法使用。

4. 现在，您的客户端请求已被批准，您必须查看添加到集群中的每台机器的服务器请求：

```
$ oc get csr
```

输出示例

```
NAME        AGE     REQUESTOR                                                   CONDITION
csr-bfd72   5m26s   system:node:ip-10-0-50-126.us-east-2.compute.internal
Pending
csr-c57lv   5m26s   system:node:ip-10-0-95-157.us-east-2.compute.internal
Pending
...
```

5. 如果剩余的 CSR 没有被批准，且处于 **Pending** 状态，请批准集群机器的 CSR：

   - 若要单独批准，请对每个有效的 CSR 运行以下命令：

     ```
     $ oc adm certificate approve <csr_name> 1
     ```

     1 **<csr_name>** 是当前 CSR 列表中 CSR 的名称。

   - 要批准所有待处理的 CSR，请运行以下命令：

     ```
     $ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}{{"\n"}}
     {{end}}{{end}}' | xargs oc adm certificate approve
     ```

6. 批准所有客户端和服务器 CSR 后，器将处于 **Ready** 状态。运行以下命令验证：

   ```
   $ oc get nodes
   ```

   输出示例

   ```
   NAME     STATUS   ROLES   AGE  VERSION
   master-0  Ready    master  73m  v1.20.0
   master-1  Ready    master  73m  v1.20.0
   master-2  Ready    master  74m  v1.20.0
   worker-0  Ready    worker  11m  v1.20.0
   worker-1  Ready    worker  11m  v1.20.0
   ```

注意

批准服务器 CSR 后可能需要几分钟时间让机器转换为 **Ready** 状态。

其他信息

- 如需有关 CSR 的更多信息，请参阅证书签名请求。

8.2.12. 初始 Operator 配置

在 control plane 初始化后，您必须立即配置一些 Operator 以便它们都可用。

先决条件

- 您的 control plane 已初始化。

流程

1. 观察集群组件上线：

   ```
   $ watch -n5 oc get clusteroperators
   ```

   输出示例

   ```
   NAME                                       VERSION AVAILABLE   PROGRESSING   DEGRADED
   ```

```
SINCE
authentication                             4.6.0   True        False         False      3h56m
cloud-credential                           4.6.0   True        False         False      29h
cluster-autoscaler                         4.6.0   True        False         False      29h
config-operator                            4.6.0   True        False         False      6h39m
console                                    4.6.0   True        False         False      3h59m
csi-snapshot-controller                    4.6.0   True        False         False      4h12m
dns                                        4.6.0   True        False         False      4h15m
etcd                                       4.6.0   True        False         False      29h
image-registry                             4.6.0   True        False         False      3h59m
ingress                                    4.6.0   True        False         False      4h30m
insights                                   4.6.0   True        False         False      29h
kube-apiserver                             4.6.0   True        False         False      29h
kube-controller-manager                    4.6.0   True        False         False      29h
kube-scheduler                             4.6.0   True        False         False      29h
kube-storage-version-migrator              4.6.0   True        False         False      4h2m
machine-api                                4.6.0   True        False         False      29h
machine-approver                           4.6.0   True        False         False      6h34m
machine-config                             4.6.0   True        False         False      3h56m
marketplace                                4.6.0   True        False         False      4h2m
monitoring                                 4.6.0   True        False         False      6h31m
network                                    4.6.0   True        False         False      29h
node-tuning                                4.6.0   True        False         False      4h30m
openshift-apiserver                        4.6.0   True        False         False      3h56m
openshift-controller-manager               4.6.0   True        False         False      4h36m
openshift-samples                          4.6.0   True        False         False      4h30m
operator-lifecycle-manager                 4.6.0   True        False         False      29h
operator-lifecycle-manager-catalog         4.6.0   True        False         False      29h
operator-lifecycle-manager-packageserver   4.6.0   True        False         False      3h59m
service-ca                                 4.6.0   True        False         False      29h
storage                                    4.6.0   True        False         False      4h30m
```

2. 配置不可用的 Operator。

8.2.12.1. 禁用默认的 OperatorHub 源

在 OpenShift Container Platform 安装过程中，默认为 OperatorHub 配置由红帽和社区项目提供的源内容的 operator 目录。在受限网络环境中，必须以集群管理员身份禁用默认目录。

流程

- 通过在 **OperatorHub** 对象中添加 **disableAllDefaultSources: true** 来禁用默认目录的源：

```
$ oc patch OperatorHub cluster --type json \
    -p '[{"op": "add", "path": "/spec/disableAllDefaultSources", "value": true}]'
```

提示

或者，您可以使用 Web 控制台管理目录源。在 **Administration → Cluster Settings → Global Configuration → OperatorHub** 页面中，点 **Sources** 选项卡，其中可创建、删除、禁用和启用单独的源。

8.2.12.2. 镜像 registry 存储配置

对于不提供默认存储的平台，Image Registry Operator 最初将不可用。安装后，您必须配置 registry 使用的存储，这样 Registry Operator 才可用。

示配置生产集群所需的持久性卷的说明。如果适用，显示有关将空目录配置为存储位置的说明，该位置只可用于非生产集群。

另外还提供了在升级过程中使用 **Recreate** rollout 策略来允许镜像 registry 使用块存储类型的说明。

8.2.12.2.1. 更改镜像 registry 的管理状态

要启动镜像 registry，需要把 Image Registry Operator 配置的 **managementState** 从 **Removed** 改为 **Managed**。

流程

- 将 **managementState** Image Registry Operator 配置从 **Removed** 改为 **Managed**。例如：

```
$ oc patch configs.imageregistry.operator.openshift.io cluster --type merge --patch '{"spec":{"managementState":"Managed"}}'
```

8.2.12.2.2. 为 IBM Power 系统配置容器镜像仓库（registry）存储

作为集群管理员，在安装后需要配置 registry 来使用存储。

先决条件

- 具有 Cluster Administrator 权限
- IBM Power 系统上的集群。
- 为集群置备的持久性存储，如 Red Hat OpenShift Container Storage。

重要

如果您只有一个副本，OpenShift Container Platform 支持对镜像 registry 存储的 **ReadWriteOnce** 访问。要部署支持高可用性的、带有两个或多个副本的镜像 registry，需要 **ReadWriteMany** 访问设置。

- 必须具有 100Gi 容量。

流程

1. 为了配置 registry 使用存储，需要修改 **configs.imageregistry/cluster** 资源中的 **spec.storage.pvc** 。

注意

使用共享存储时，请查看您的安全设置以防止被外部访问。

2. 验证您没有 registry pod:

```
$ oc get pod -n openshift-image-registry
```

注意

如果存储类型为 **emptyDIR**，则副本数不能超过 **1**。

3. 检查 registry 配置：

```
$ oc edit configs.imageregistry.operator.openshift.io
```

输出示例

```
storage:
  pvc:
    claim:
```

将 **claim** 字段留空以允许自动创建一个 **image-registry-storage** PVC。

4. 检查 **clusteroperator** 的状态：

```
$ oc get clusteroperator image-registry
```

5. 确保您的 registry 设置为 manage，以启用镜像的构建和推送。

- 运行：

```
$ oc edit configs.imageregistry/cluster
```

然后将行改

```
managementState: Removed
```

为

```
managementState: Managed
```

8.2.12.2.3. 在非生产集群中配置镜像 registry 存储

您必须为 Image Registry Operator 配置存储。对于非生产集群，您可以将镜像 registry 设置为空目录。如果您这样做，重启 registry 后会丢失所有镜像。

流程

- 将镜像 registry 存储设置为空目录：

```
$ oc patch configs.imageregistry.operator.openshift.io cluster --type merge --patch '{"spec":
{"storage":{"emptyDir":{}}}}'
```

警告

仅可为非生产集群配置这个选项。

如果在 Image Registry Operator 初始化其组件前运行此命令，**oc patch** 命令会失败并显示以下错误：

```
Error from server (NotFound): configs.imageregistry.operator.openshift.io "cluster" not found
```

等待几分钟，然后再次运行该命令。

8.2.13. 在用户置备的基础架构上完成安装

完成 Operator 配置后，可以在您提供的基础架构上完成集群安装。

先决条件

- 您的 control plane 已初始化。
- 已完成初始 Operator 配置。

流程

1. 使用以下命令确认所有集群组件都已在线：

```
$ watch -n5 oc get clusteroperators
```

输出示例

```
NAME                                       VERSION AVAILABLE   PROGRESSING   DEGRADED
SINCE
authentication                             4.6.0   True        False         False     3h56m
cloud-credential                           4.6.0   True        False         False     29h
cluster-autoscaler                         4.6.0   True        False         False     29h
config-operator                            4.6.0   True        False         False     6h39m
console                                    4.6.0   True        False         False     3h59m
csi-snapshot-controller                    4.6.0   True        False         False     4h12m
dns                                        4.6.0   True        False         False     4h15m
etcd                                       4.6.0   True        False         False     29h
image-registry                             4.6.0   True        False         False     3h59m
ingress                                    4.6.0   True        False         False     4h30m
insights                                   4.6.0   True        False         False     29h
kube-apiserver                             4.6.0   True        False         False     29h
kube-controller-manager                    4.6.0   True        False         False     29h
kube-scheduler                             4.6.0   True        False         False     29h
kube-storage-version-migrator              4.6.0   True        False         False     4h2m
machine-api                                4.6.0   True        False         False     29h
machine-approver                           4.6.0   True        False         False     6h34m
machine-config                             4.6.0   True        False         False     3h56m
```

```
marketplace                                4.6.0   True        False         False      4h2m
monitoring                                 4.6.0   True        False         False      6h31m
network                                    4.6.0   True        False         False      29h
node-tuning                                4.6.0   True        False         False      4h30m
openshift-apiserver                        4.6.0   True        False         False      3h56m
openshift-controller-manager               4.6.0   True        False         False      4h36m
openshift-samples                          4.6.0   True        False         False      4h30m
operator-lifecycle-manager                 4.6.0   True        False         False      29h
operator-lifecycle-manager-catalog         4.6.0   True        False         False      29h
operator-lifecycle-manager-packageserver   4.6.0   True        False         False      3h59m
service-ca                                 4.6.0   True        False         False      29h
storage                                    4.6.0   True        False         False      4h30m
```

或者，通过以下命令，如果所有集群都可用您会接到通知。它还检索并显示凭证：

```
$ ./openshift-install --dir <installation_directory> wait-for install-complete 1
```

1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

输出示例

```
INFO Waiting up to 30m0s for the cluster to initialize...
```

Cluster Version Operator 完成从 Kubernetes API 服务器部署 OpenShift Container Platform 集群时，命令运行成功。

重要

- 安装程序生成的 Ignition 配置文件包含在 24 小时后过期的证书，然后在过期时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外情况是，您需要手动批准待处理的 **node-bootstrapper** 证书签名请求（CSR）来恢复 kubelet 证书。如需更多信息，请参阅*从过期的 control plane 证书中恢复*的文档。
- 建议您在生成 12 小时后使用 Ignition 配置文件，因为集群安装后 24 小时证书从 16 小时轮转至 22 小时。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中运行证书更新时避免安装失败。

2. 确认 Kubernetes API 服务器正在与 pod 通信。

   a. 要查看所有 pod 的列表，请使用以下命令：

   ```
   $ oc get pods --all-namespaces
   ```

   输出示例

   ```
   NAMESPACE                       NAME                                         READY   STATUS
   RESTARTS   AGE
   openshift-apiserver-operator    openshift-apiserver-operator-85cb746d55-zqhs8   1/1
   Running   1          9m
   openshift-apiserver             apiserver-67b9g                               1/1     Running   0
   3m
   openshift-apiserver             apiserver-ljcmx                               1/1     Running   0
   ```

```
1m
openshift-apiserver                apiserver-z25h4                                  1/1     Running     0
2m
openshift-authentication-operator authentication-operator-69d5d8bf84-vh2n8         1/1
Running     0          5m
...
```

b. 使用以下命令，查看上一命令的输出中所列 pod 的日志：

```
$ oc logs <pod_name> -n <namespace>
```
(1)

(1) 指定 pod 名称和命名空间，如上一命令的输出中所示。

如果 pod 日志显示，Kubernetes API 服务器可以与集群机器通信。

8.2.14. OpenShift Container Platform 的 Telemetry 访问

在 OpenShift Container Platform 4.6 中，默认运行的 Telemetry 服务提供有关集群健康状况和成功更新的指标，需要访问互联网。如果您的集群连接到互联网，Telemetry 会自动运行，而且集群会注册到 OpenShift Cluster Manager。

确认 OpenShift Cluster Manager 清单正确后，可以由 Telemetry 自动维护，也可以使用 OpenShift Cluster Manager 手动维护，使用订阅监控来跟踪帐户或多集群级别的 OpenShift Container Platform 订阅。

其他资源

- 有关 Telemetry 服务的更多信息，请参阅关于远程健康监控。

8.2.15. 后续步骤

- 自定义集群。
- 如果您用来安装集群的镜像 registry 具有一个可信任的 CA，通过配置额外的信任存储将其添加到集群中。

第 9 章 在 OPENSTACK 上安装

9.1. 使用自定义配置在 OPENSTACK 上安装集群

在 OpenShift Container Platform 版本 4.6 中，您可以在 Red Hat OpenStack Platform（RHOSP）上安装自定义集群。要自定义安装，请在安装集群前修改 **install-config.yaml** 中的参数。

9.1.1. 先决条件

- 查看有关 OpenShift Container Platform 安装和更新流程的详细信息。
  - 在 *Available platforms* 部分验证 OpenShift Container Platform 4.6 是否与您的 RHOSP 版本兼容。您还可以查看 OpenShift Container Platform 在 RHOSP 中的支持 来比较不同版本的平台支持。
- 验证您的网络配置不依赖于供应商网络。不支持提供商网络。
- 在 RHOSP 中安装了存储服务，如快存储（Cinder）或对象存储（Swift）。对象存储是 OpenShift Container Platform registry 集群部署的推荐存储技术。如需更多信息，请参阅 优化存储。
- 在 RHOSP 中启用了元数据服务

9.1.2. 在 RHOSP 上安装 OpenShift Container Platform 的资源指南

您的 Red Hat OpenStack Platform（RHOSP）配额需要满足以下条件才支持 OpenShift Container Platform 安装：

表 9.1. RHOSP 上默认 OpenShift Container Platform 集群的建议资源

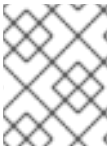
资源	值
浮动 IP 地址	3
端口	15
路由器	1
子网	1
RAM	112 GB
vCPUs	28
卷存储	275 GB
实例	7
安全组	3

资源	值
安全组规则	60

集群或许能使用少于推荐数量的资源来运作，但其性能无法保证。

重要

如果 RHOSP 对象存储（Swift）可用，并由具有 **swiftoperator** 角色的用户帐户执行，它会作为 OpenShift Container Platform 镜像 registry 的默认后端。在这种情况下，卷存储需要有 175GB。根据镜像 registry 的大小，Swift 空间要求会有所不同。



注意

默认情况下，您的安全组和安全组规则配额可能较低。如果遇到问题，请以 admin 的身份运行 **openstack quota set --secgroups 3 --secgroup-rules 60 <project>** 来提高配额。

OpenShift Container Platform 部署由 control plane 机器、计算机器和 bootstrap 机器组成。

9.1.2.1. control plane 机器

默认情况下，OpenShift Container Platform 安装过程会创建三台 control plane 机器。

每台机器都需要：

- 来自 RHOSP 配额的实例
- 来自 RHOSP 配额的端口
- 至少 16 GB 内存、4 个 vCPU 和 100 GB 存储空间的类别

9.1.2.2. 计算机器

默认情况下，OpenShift Container Platform 安装过程会创建三台计算机器。

每台机器都需要：

- 来自 RHOSP 配额的实例
- 来自 RHOSP 配额的端口
- 至少有 8 GB 内存、2 个 vCPU 和 100 GB 存储空间的类别

提示

计算机器托管您在 OpenShift Container Platform 上运行的应用程序；运行数量应尽可能多。

9.1.2.3. bootstrap 机器

在安装时，会临时置备 bootstrap 机器来支持 control plane。生产控制平面就绪后，bootstrap 机器会被取消置备。

bootstrap 机器需要：

- 来自 RHOSP 配额的实例
- 来自 RHOSP 配额的端口
- 至少 16 GB 内存、4 个 vCPU 和 100 GB 存储空间的类别

9.1.3. OpenShift Container Platform 的互联网访问

在 OpenShift Container Platform 4.6 中，您需要访问互联网来安装集群。

您必须具有以下互联网访问权限：

- 访问 OpenShift Cluster Manager 以下载安装程序并执行订阅管理。如果集群可以访问互联网，并且没有禁用 Telemetry，该服务会自动授权您的集群。
- 访问 Quay.io，以获取安装集群所需的软件包。
- 获取执行集群更新所需的软件包。

重要

如果您的集群无法直接访问互联网，则可以在置备的某些类基础架构上执行受限网络安装。在此过程中，您要下载所需的内容，并使用它在镜像 registry（mirror registry） 中填充安装集群并生成安装程序所需的软件包。对于某些安装类型，集群要安装到的环境不需要访问互联网。在更新集群之前，要更新 registry 镜像系统中的内容。

9.1.4. 在 RHOSP 上启用 Swift

Swift 由具有 **swiftoperator** 角色的用户帐户操控。在运行安装程序前，将该角色添加到帐户。

重要

如果 Red Hat OpenStack Platform（RHOSP） 对象存储服务 （通常称为 Swift）可用，OpenShift Container Platform 会使用它作为镜像 registry 存储。如果无法使用，安装程序将依赖于 RHOSP 块存储服务，通常称为 Cinder 。

如果 Swift 存在且您想要使用 Swift，则必须启用对其的访问。如果不存在，或者您不想使用它，请跳过这个部分。

先决条件

- 在目标环境中具有 RHOSP 管理员帐户
- 已安装 Swift 服务。
- 在Ceph RGW 上启用了**account in url** 选项。

流程

在 RHOSP 上启用 Swift：

1. 在 RHOSP CLI 中以管理员身份，将 **swiftoperator** 角色添加到要访问 Swift 的帐户：

```
$ openstack role add --user <user> --project <project> swiftoperator
```

-

您的 RHOSP 部署现可以使用 Swift 用于镜像 registry。

9.1.5. 验证外部网络访问

OpenShift Container Platform 安装进程需要外部网络访问权限。您必须为其提供外部网络值，否则部署会失败。在运行安装进程前，请验证 Red Hat OpenStack Platform (RHOSP) 中是否存在具有外部路由器类型的网络。

先决条件

- 将 OpenStack 联网服务配置为使用 DHCP 代理转发实例 DNS 查询

流程

1. 使用 RHOSP CLI 验证"外部"网络的名称和 ID：

```
$ openstack network list --long -c ID -c Name -c "Router Type"
```

输出示例

```
+--------------------------------------+---------------+-------------+
| ID                                   | Name          | Router Type |
+--------------------------------------+---------------+-------------+
| 148a8023-62a7-4672-b018-003462f8d7dc | public_network | External    |
+--------------------------------------+---------------+-------------+
```

网络列表中会显示具有外部路由器类型的网络。如果最少有一个没有，请参阅 创建默认浮动 IP 网络和创建默认供应商网络。

重要

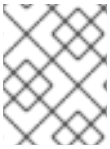
如果外部网络 CIDR 范围与某一个默认网络范围重叠，您必须在运行安装进程前更改 **install-config.yaml** 文件中匹配的网络范围。

默认的网络范围：

网络	范围
machineNetwork	10.0.0.0/16
serviceNetwork	172.30.0.0/16
clusterNetwork	10.128.0.0/14

警告

如果安装程序找到多个同名的镜像，它会随机设置其中之一。为避免这种行为，请在 RHOSP 中为资源创建唯一名称。



注意

如果启用了 Neutron 中继服务插件，则默认创建中继端口。如需更多信息，请参阅 Neutron 中继端口。

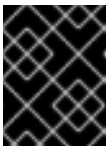
9.1.6. 为安装程序定义参数

OpenShift Container Platform 安装程序依赖于一个名为 **clouds.yaml** 的文件。该文件描述了 Red Hat OpenStack Platform (RHOSP) 配置参数，包括项目名称、登录信息和授权服务 URL。

流程

1. 创建 **clouds.yaml** 文件：

   - 如果您的 RHOSP 发行版包含 Horizon web UI，请在该 UI 中生成 **clouds.yaml** 文件。

     重要

     请记住在 **auth** 字段中添加密码。您也可以把 secret 保存在 **clouds.yaml** 以外的一个独立的文件中。

   - 如果您的 RHOSP 发行版不包含 Horizon Web UI，或者您不想使用 Horizon，请自行创建该文件。如需有关 **clouds.yaml** 的详细信息，请参阅 RHOSP 文档中的配置文件。

     ```
     clouds:
       shiftstack:
         auth:
           auth_url: http://10.10.14.42:5000/v3
           project_name: shiftstack
           username: shiftstack_user
           password: XXX
           user_domain_name: Default
           project_domain_name: Default
       dev-env:
         region_name: RegionOne
         auth:
           username: 'devuser'
           password: XXX
           project_name: 'devonly'
           auth_url: 'https://10.10.14.22:5001/v2.0'
     ```

2. 如果您的 RHOSP 安装使用自签名证书颁发机构 (CA) 证书进行端点身份验证：

   a. 将 CA 文件复制到您的机器中。

b. 将机器添加到证书颁发机构信任捆绑包中：

```
$ sudo cp ca.crt.pem /etc/pki/ca-trust/source/anchors/
```

c. 更新信任捆绑包：

```
$ sudo update-ca-trust extract
```

d. 将 **cacerts** 键添加到 **clouds.yaml** 文件。该值必须是到 CA 证书的绝对路径，则其可以被非根用户访问：

```
clouds:
  shiftstack:
    ...
    cacert: "/etc/pki/ca-trust/source/anchors/ca.crt.pem"
```

提示

使用自定义 CA 证书运行安装程序后，您可以通过编辑 **cloud-provider-config** keymap 中的**ca-cert.pem** 键的值来更新证书。在命令行中运行：

```
$ oc edit configmap -n openshift-config cloud-provider-config
```

3. 将 **clouds.yaml** 文件放在以下位置之一：

a. **OS_CLIENT_CONFIG_FILE** 环境变量的值

b. 当前目录

c. 特定于 Unix 的用户配置目录，如 **~/.config/openstack/clouds.yaml**

d. 特定于 Unix 的站点配置目录，如 **/etc/openstack/clouds.yaml**
安装程序会按照以上顺序搜索 **clouds.yaml**。

9.1.7. 获取安装程序

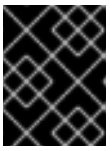
在安装 OpenShift Container Platform 之前，将安装文件下载到本地计算机上。

先决条件

- 运行 Linux 或 macOS 的计算机，本地磁盘空间为 500 MB

流程

1. 访问 OpenShift Cluster Manager 站点的 Infrastructure Provider 页面。如果您有红帽帐号，请使用自己的凭证登录。如果没有，请创建一个帐户。
2. 选择您的基础架构供应商。
3. 进入适用于您的安装类型的页面，下载您的操作系统的安装程序，并将文件放在要保存安装配置文件的目录中。。



重要

安装程序会在用来安装集群的计算机上创建若干文件。在完成集群安装后，您必须保留安装程序和安装程序所创建的文件。这两个文件都需要删除集群。

重要

删除安装程序创建的文件不会删除您的集群，即使集群在安装过程中失败也是如此。要删除集群，为特定云供应商完成 OpenShift Container Platform 卸载流程。

4. 提取安装程序。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ tar xvf openshift-install-linux.tar.gz
```

5. 从 Red Hat OpenShift Cluster Manager 下载安装 pull secret 。通过此 pull secret，您可以进行所含授权机构提供的服务的身份验证，这些服务包括为 OpenShift Container Platform 组件提供容器镜像的 Quay.io。

9.1.8. 创建安装配置文件

您可以自定义在 Red Hat OpenStack Platform（RHOSP）上安装的 OpenShift Container Platform 集群。

先决条件

- 获取 OpenShift Container Platform 安装程序以及集群的 pull secret。

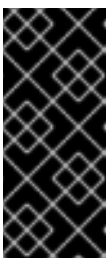
流程

1. 创建 **install-config.yaml** 文件。

   a. 更改到包含安装程序的目录，再运行以下命令：

```
$ ./openshift-install create install-config --dir <installation_directory> 1
```

   1 对于 **<installation_directory>**，请指定用于保存安装程序所创建的文件的目录名称。



重要

指定一个空目录。一些安装信息，如 bootstrap X.509 证书，有较短的过期间隔，因此不要重复使用安装目录。如果要重复使用另一个集群安装中的个别文件，可以将其复制到您的目录中。但是，一些安装数据的文件名可能会在发行版本之间有所改变。从 OpenShift Container Platform 老版本中复制安装文件时要格外小心。

   b. 在提示符处，提供您的云的配置详情：

      i. 可选：选择用来访问集群机器的 SSH 密钥。

注意

对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 **ssh-agent** 进程使用的 SSH 密钥。

ii. 选择 **openstack** 作为目标平台。

iii. 指定用于安装集群的 Red Hat OpenStack Platform (RHOSP) 外部网络名称。

iv. 指定用于从外部访问 OpenShift API 的浮动 IP 地址。

v. 指定至少有 16 GB RAM 用于 control plane 节点，以及计算节点的 8 GB RAM。

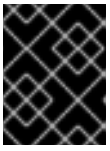
vi. 选择集群要部署到的基域。所有 DNS 记录都将是这个基域的子域，并包含集群名称。

vii. 为集群输入一个名称。名称不能多于 14 个字符。

viii. 粘贴 Red Hat OpenShift Cluster Manager 中的 pull secret 。

2. 修改 **install-config.yaml** 文件。您可以在**安装配置参数**部分中找到有关可用参数的更多信息。

3. 备份 **install-config.yaml** 文件，以便用于安装多个集群。



重要

install-config.yaml 文件会在安装过程中消耗掉。如果要重复使用此文件，必须现在备份。

其他资源

有关可用参数的更多信息，请参阅安装配置参数部分。

9.1.8.1. 在安装过程中配置集群范围代理

生产环境可能会拒绝直接访问互联网，而是提供 HTTP 或 HTTPS 代理。您可以通过在 **install-config.yaml** 文件中配置代理设置，将新的 OpenShift Container Platform 集群配置为使用代理。

先决条件

- 您有一个现有的 **install-config.yaml** 文件。
- 您检查了集群需要访问的站点，并决定是否需要绕过代理。默认情况下代理所有集群出口流量，包括对托管云供应商 API 的调用。您需要将站点添加到 **Proxy** 对象的 **spec.noProxy** 字段来绕过代理。

注意

Proxy 对象 **status.noProxy** 字段使用安装配置中的 **networking.machineNetwork[].cidr**、**networking.clusterNetwork[].cidr** 和 **networking.serviceNetwork[]** 字段的值填充。

对于在 Amazon Web Services(AWS)、Google Cloud Platform(GCP)、Microsoft Azure 和 Red Hat OpenStack Platform(RHOSP)上安装, **Proxy** 对象 **status.noProxy** 字段也会使用实例元数据端点填充(**169.254.169.254**)。

流程

1. 编辑 **install-config.yaml** 文件并添加代理设置。例如：

```
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: example.com 3
additionalTrustBundle: | 4
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
...
```

1. 用于创建集群外 HTTP 连接的代理 URL。URL 必须是 **http**。
2. 用于创建集群外 HTTPS 连接的代理 URL。
3. 要排除在代理中的目标域名、IP 地址或其他网络 CIDR 的逗号分隔列表。在域前面加 **.** 来仅匹配子域。例如： **.y.com** 匹配 **x.y.com**，但不匹配 **y.com**。使用 * 绕过所有目的地的代理。
4. 如果提供，安装程序会在 **openshift-config 命名空间中生成名为 user-ca- bundle** 的配置映射来保存额外的 CA 证书。如果您提供 **additionalTrustBundle** 和至少一个代理设置，则 **Proxy** 对象会被配置为引用 **trustedCA** 字段中的 **user-ca-bundle** 配置映射。然后，Cluster Network Operator 会创建一个 **trusted-ca-bundle** 配置映射，该配置映射将为 **trustedCA** 参数指定的内容与 RHCOS 信任捆绑包合并。 **additionalTrustBundle** 字段是必需的，除非代理的身份证书由来自 RHCOS 信任捆绑包的颁发机构签名。

注意

安装程序不支持代理的 **readinessEndpoints** 字段。

2. 保存该文件，并在安装 OpenShift Container Platform 时引用。

安装程序会创建一个名为 **cluster** 的集群范围代理，该代理使用提供的 **install-config.yaml** 文件中的代理设置。如果没有提供代理设置，仍然会创建一个 **cluster Proxy** 对象，但它会有一个空 **spec**。

注意

只支持名为 **cluster** 的 **Proxy** 对象，且无法创建额外的代理。

9.1.9. 安装配置参数

在部署 OpenShift Container Platform 集群前，您可以提供参数值，以描述托管集群的云平台的帐户并选择性地自定义集群平台。在创建 **install-config.yaml** 安装配置文件时，您可以通过命令行来提供所需的参数的值。如果要自定义集群，可以修改 **install-config.yaml** 文件来提供关于平台的更多信息。

注意

安装之后，您无法修改 **install-config.yaml** 文件中的这些参数。

重要

openshift-install 命令不验证参数的字段名称。如果指定了不正确的名称，则不会创建相关的文件或对象，且不会报告错误。确保所有指定的参数的字段名称都正确。

9.1.9.1. 所需的配置参数

下表描述了所需的安装配置参数：

表 9.2. 所需的参数

参数	描述	值
apiVersion	**install-config.yaml** 内容的 API 版本。当前版本是 **v1**。安装程序还可能支持旧的 API 版本。	字符串
baseDomain	云供应商的基域。此基础域用于创建到 OpenShift Container Platform 集群组件的路由。集群的完整 DNS 名称是 **baseDomain** 和 **metadata.name** 参数值的组合，其格式为 **<metadata.name>.<baseDomain>**。	完全限定域名或子域名，如 **example.com**。
metadata	Kubernetes 资源 **ObjectMeta**，其中只消耗 **name** 参数。	对象
metadata.name	集群的名称。集群的 DNS 记录是 **{{.metadata.name}}.{{.baseDomain}}** 的子域。	小写字母,连字符(-)和句点(.)的字符串，如 **dev**。该字符串长度必须为 14 个字符或更少。
platform	执行安装的具体平台配置：**aws**、**baremetal**、**azure**、**openstack**、**ovirt**、**vsphere**。有关 **platform.<platform>** 参数的额外信息，请参考下表来了解您的具体平台。	对象

参数	描述	值

pullSecret	从 Red Hat OpenShift Cluster Manager 获取 pull secret，验证从 Quay.io 等服务中下载 OpenShift Container Platform 组件的容器镜像。	<pre>{
		  "auths":{
		  "cloud.openshift.com":{
		  "auth":"b3Blb=",
		  "email":"you@example.com"
		  },
		  "quay.io":{
		  "auth":"b3Blb=",
		  "email":"you@example.com"
		  }
		  }
		}</pre>

9.1.9.2. 网络配置参数

您可以根据现有网络基础架构的要求自定义安装配置。例如，您可以扩展集群网络的 IP 地址块，或者提供不同于默认值的不同 IP 地址块。

只支持 IPv4 地址。

表 9.3. 网络参数

参数	描述	值
networking	集群网络的配置。	对象 **注意** 您不能在安装后修改 **networking** 对象指定的参数。
networking.networkType	要安装的集群网络供应商 Container Network Interface（CNI）插件。	**OpenShiftSDN** 或 **OVNKubernetes**。默认值为 **OpenShiftSDN**。

networking.clusterNetwork	pod 的 IP 地址块。
	
	默认值为 **10.128.0.0/14**，主机前缀为 **/23**。
	
	如果您指定多个 IP 地址块，则块不得互相重叠。
		一个对象数组。例如：
		
		<pre>networking:
		  clusterNetwork:
		  - cidr: 10.128.0.0/14
		  hostPrefix: 23</pre>

networking.clusterNetwork.cidr	使用 **networking.clusterNetwork** 时需要此项。IP 地址块。 一个 IPv4 网络。	使用 CIDR 形式的 IP 地址块。IPv4 块的前缀长度介于 **0** 到 **32** 之间。

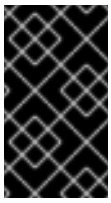
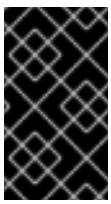
参数	描述	值
networking.clusterNetwork.hostPrefix	分配给每个单独节点的子网前缀长度。例如，如果 **hostPrefix** 设为 **23**，则每个节点从所给的 **cidr** 中分配一个 **/23** 子网。**hostPrefix** 值 **23** 提供 510（2^(32 - 23)- 2）个 pod IP 地址。	子网前缀。 默认值为 **23**。
networking.serviceNetwork	服务的 IP 地址块。默认值为 **172.30.0.0/16**。 OpenShift SDN 和 OVN-Kubernetes 网络供应商只支持服务网络的一个 IP 地址块。	CIDR 格式具有 IP 地址块的数组。例如： networking: serviceNetwork: - 172.30.0.0/16
networking.machineNetwork	机器的 IP 地址块。 如果您指定多个 IP 地址块，则块不得互相重叠。	一个对象数组。例如： networking: machineNetwork: - cidr: 10.0.0.0/16
networking.machineNetwork.cidr	使用 **networking.machineNetwork** 时需要。IP 地址块。libvirt 以外的所有平台的默认值为 **10.0.0.0/16**。对于 libvirt，默认值为 **192.168.126.0/24**。	CIDR 表示法中的 IP 网络块。 例如：**10.0.0.0/16**。 **注意** 将 **networking.machineNetwork** 设置为与首选 NIC 所在的 CIDR 匹配。

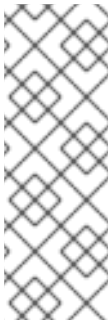
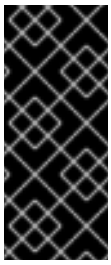
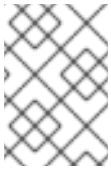
9.1.9.3. 可选配置参数

下表描述了可选安装配置参数：

表 9.4. 可选参数

参数	描述	值
additionalTrustBundle	添加到节点可信证书存储中的 PEM 编码 X.509 证书捆绑包。配置了代理时，也可以使用这个信任捆绑包。	字符串
compute	组成计算节点的机器的配置。	machine-pool 对象的数组。详情请查看以下"Machine-pool"表。

参数	描述	值
compute.architecture	决定池中机器的指令集合架构。目前不支持异构集群，因此所有池都必须指定相同的架构。有效值为 **amd64** （默认值）。	字符串
compute.hyperthreading	是否在计算机器上启用或禁用并发多线程或**超线程**。默认情况下，启用并发多线程以提高机器内核的性能。  **重要** 如果禁用并发多线程，请确保在容量规划时考虑到机器性能可能会显著降低的问题。	**Enabled** 或 **Disabled**
compute.name	使用 **compute** 时需要此值。机器池的名称。	**worker**
compute.platform	使用 **compute** 时需要此值。使用此参数指定托管 worker 机器的云供应商。此参数值必须与 **controlPlane.platform** 参数值匹配。	**aws**、**azure**、**gcp**、**openstack**、**ovirt**、**vsphere** 或 **{}**
compute.replicas	要置备的计算机器数量，也称为 worker 机器。	大于或等于 **2** 的正整数。默认值为 **3**。
controlPlane	组成 control plane 的机器的配置。	**MachinePool** 对象的数组。详情请查看以下"Machine-pool"表。
controlPlane.architecture	决定池中机器的指令集合架构。目前不支持异构集群，因此所有池都必须指定相同的架构。有效值为 **amd64** （默认值）。	字符串
controlPlane.hyperthreading	是否在 control plane 机器上启用或禁用并发多线程或**超线程**。默认情况下，启用并发多线程以提高机器内核的性能。  **重要** 如果禁用并发多线程，请确保在容量规划时考虑到机器性能可能会显著降低的问题。	**Enabled** 或 **Disabled**

参数	描述	值
controlPlane.name	使用 **controlPlane** 时需要。机器池的名称。	**master**
controlPlane.platform	使用 **controlPlane** 时需要。使用此参数指定托管 control plane 机器的云供应商。此参数值必须与 **compute.platform** 参数值匹配。	**aws**、**azure**、**gcp**、**openstack**、**ovirt**、**vsphere** 或 **{}**
controlPlane.replicas	要置备的 control plane 机器数量。	唯一支持的值是 **3**，它是默认值。
credentialsMode	Cloud Credential Operator（CCO）模式。如果没有指定任何模式，CCO 会动态地尝试决定提供的凭证的功能，在支持多个模式的平台上使用 mint 模式。  **注意** 不是所有 CCO 模式都支持所有云供应商。如需有关 CCO 模式的更多信息，请参阅*Red Hat Operator 参考指南*内容中的 *Cloud Credential Operator* 条目。	**Mint**、**Passthrough**、**Manual** 或空字符串(**""**)。
fips	启用或禁用 FIPS 模式。默认为 **false**（禁用）。如果启用了 FIPS 模式，运行 OpenShift Container Platform 的 Red Hat Enterprise Linux CoreOS (RHCOS) 机器会绕过默认的 Kubernetes 加密套件，并使用由 RHCOS 提供的加密模块。  **重要** 只有在 **x86_64** 架构中的 OpenShift Container Platform 部署支持 FIPS 验证的/Modules in Process 加密库。  **注意** 如果使用 Azure File 存储，则无法启用 FIPS 模式。	**false** 或 **true**

参数	描述	值
imageContentSources	release-image 内容的源和仓库。	对象数组。包括一个 **source** 以及可选的 **mirrors**，如下表所示。
imageContentSources.source	使用 **imageContentSources** 时需要。指定用户在镜像拉取规格中引用的仓库。	字符串
imageContentSources.mirrors	指定可能还包含同一镜像的一个或多个仓库。	字符串数组
publish	如何发布或公开集群的面向用户的端点，如 Kubernetes API、OpenShift 路由。	**Internal** 或 **External**。默认值为 **External**。 在非云平台上不支持将此字段设置为 **Internal**。  **重要** 如果将字段的值设为 **Internal**，集群将无法运行。如需更多信息，请参阅 BZ#1953035。
sshKey	用于验证集群机器访问的 SSH 密钥或密钥。  **注意** 对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 **ssh-agent** 进程使用的 SSH 密钥。	一个或多个密钥。例如： sshKey: <key1> <key2> <key3>

9.1.9.4. 其他 Red Hat OpenStack Platform（RHOSP）配置参数

下表描述了其他 RHOSP 配置参数：

表 9.5. 其他 RHOSP 参数

参数	描述	值
compute.platform.openstack.rootVolume.size	对于计算机器，以 GB 为单位表示的根卷大小。如果您不设置这个值，机器将使用临时存储。	整数，如 **30**。
compute.platform.openstack.rootVolume.type	对于计算机器，根卷的类型。	字符串，如 **performance**。
controlPlane.platform.openstack.rootVolume.size	对于 control plane 机器，以 GB 为单位表示的根卷大小。如果您不设置这个值，机器将使用临时存储。	整数，如 **30**。
controlPlane.platform.openstack.rootVolume.type	对于 control plane 机器，根卷的类型。	字符串，如 **performance**。
platform.openstack.cloud	要使用的 RHOSP 云的名称，来自于 **clouds.yaml** 文件中的云列表。	字符串，如 **MyCloud**。
platform.openstack.externalNetwork	用于安装的 RHOSP 外部网络名称。	字符串，如 **external**。
platform.openstack.computeFlavor	用于 control plane 和计算机器的 RHOSP 类别。	字符串，如 **m1.xlarge**。

9.1.9.5. 可选 RHOSP 配置参数

下表描述了可选 RHOSP 配置参数：

表 9.6. 可选的 RHOSP 参数

参数	描述	值
compute.platform.openstack.additionalNetworkIDs	与计算机器关联的其他网络。不能为额外网络创建允许的地址对。	一个或多个 UUID 列表作为字符串。例如：**fa806b2f-ac49-4bce-b9db-124bc64209bf**。

参数	描述	值
compute.platform.openstack.additionalSecurityGroupIDs	与计算机器关联的其他安全组。	一个或多个 UUID 列表作为字符串。例如：**7ee219f3-d2e9-48a1-96c2-e7429f1b0da7**。
compute.platform.openstack.zones	RHOSP Compute（Nova）可用区（AZ）在其中安装机器。如果没有设置此参数，安装程序会依赖于配置了 RHOSP 管理员的 Nova 的默认设置。 在使用 Kuryr 的集群上，RHOSP Octavia 不支持可用域。负载均衡器，如果您使用 Amphora 供应商驱动程序，则依赖 Amphora 虚拟机的 OpenShift Container Platform 服务不会根据此属性的值创建。	字符串列表。例如：**["zone-1", "zone-2"]**。
controlPlane.platform.openstack.additionalNetworkIDs	与 control plane 机器关联的额外网络。不能为额外网络创建允许的地址对。	一个或多个 UUID 列表作为字符串。例如：**fa806b2f-ac49-4bce-b9db-124bc64209bf**。
controlPlane.platform.openstack.additionalSecurityGroupIDs	与 control plane 机器关联的其他安全组。	一个或多个 UUID 列表作为字符串。例如：**7ee219f3-d2e9-48a1-96c2-e7429f1b0da7**。
controlPlane.platform.openstack.zones	RHOSP Compute（Nova）可用区（AZ）在其中安装机器。如果没有设置此参数，安装程序会依赖于配置了 RHOSP 管理员的 Nova 的默认设置。 在使用 Kuryr 的集群上，RHOSP Octavia 不支持可用域。负载均衡器，如果您使用 Amphora 供应商驱动程序，则依赖 Amphora 虚拟机的 OpenShift Container Platform 服务不会根据此属性的值创建。	字符串列表。例如：**["zone-1", "zone-2"]**。

参数	描述	值
platform.openstack.clusterOSImage	安装程序从中下载 RHCOS 镜像的位置。 您必须设置此参数以便在受限网络中执行安装。	HTTP 或 HTTPS URL，可选使用 SHA-256 checksum。 例如：**http://mirror.example.com/images/rhcos-43.81.201912131630.0-openstack.x86_64.qcow2.gz?sha256=ffebbd68e8a1f2a245ca19522c16c86f67f9ac8e4e0c1f0a812b068b16f7265d**。该值也可以是现有 Glance 镜像的名称，如 **my-rhcos**。

platform.openstack.defaultMachinePlatform	默认机器池平台配置。	<pre>{
		  "type": "ml.large",
		  "rootVolume": {
		  "size": 30,
		  "type": "performance"
		  }
		}</pre>

platform.openstack.ingressFloatingIP	与 Ingress 端口关联的现有浮动 IP 地址。要使用此属性，还必须定义 **platform.openstack.externalNetwork** 属性。	IP 地址，如 **128.0.0.1**。
platform.openstack.lbFloatingIP	与 API 负载均衡器关联的现有浮动 IP 地址。要使用此属性，还必须定义 **platform.openstack.externalNetwork** 属性。	IP 地址，如 **128.0.0.1**。
platform.openstack.externalDNS	集群实例用于进行 DNS 解析的外部 DNS 服务器的 IP 地址。	一个 IP 地址列表作为字符串。例如，**["8.8.8.8", "192.168.1.12"]**。

参数	描述	值
platform.openstack.machinesSubnet	集群节点使用的 RHOSP 子网的 UUID。在这个子网上创建节点和虚拟 IP（VIP）端口。 **networking.machineNetwork** 中的第一个项需要和 **machinesSubnet** 的值匹配。 如果部署到自定义子网中，则无法将外部 DNS 服务器指定到 OpenShift Container Platform 安装程序。反之，把 DNS 添加到 RHOSP 的子网。	作为字符串的 UUID。例如： **fa806b2f-ac49-4bce-b9db-124bc64209bf**。

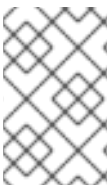
9.1.9.6. RHOSP 部署中的自定义子网

另外，您还可以在您选择的 Red Hat OpenStack Platform（RHOSP）子网中部署集群。子网的 GUID 作为 **install-config.yaml** 文件中的 **platform.openstack.machinesSubnet** 的值传递。

此子网被用作集群的主子网，在其上创建节点和端口。

在使用自定义子网运行 OpenShift Container Platform 安装程序前，请验证：

- 目标网络和子网可用。
- 目标子网上启用了 DHCP。
- 您可提供在目标网络上有创建端口权限的安装程序凭证。
- 如果您的网络配置需要一个路由器，它会在 RHOSP 中创建。有些配置依赖于路由器来转换浮动 IP 地址。
- 您的网络配置不依赖于供应商网络。不支持提供商网络。



注意

默认情况下，API VIP 使用 x.x.x.5，Ingress VIP 从网络 CIDR 块获取 x.x.x.7。要覆盖这些默认值，为 DHCP 分配池以外的 **platform.openstack.apiVIP** 和 **platform.openstack.ingressVIP** 设置值。

9.1.9.7. RHOSP 的自定义 install-config.yaml 文件示例

此示例 **install-config.yaml** 展示了所有可能的 Red Hat OpenStack Platform (RHOSP) 自定义选项。

重要

此示例文件仅供参考。您必须使用安装程序来获取 **install-config.yaml** 文件。

```
apiVersion: v1
baseDomain: example.com
```

```
clusterID: os-test
controlPlane:
  name: master
  platform: {}
  replicas: 3
compute:
- name: worker
  platform:
    openstack:
      type: ml.large
  replicas: 3
metadata:
  name: example
networking:
  clusterNetwork:
  - cidr: 10.128.0.0/14
    hostPrefix: 23
  machineNetwork:
  - cidr: 10.0.0.0/16
  serviceNetwork:
  - 172.30.0.0/16
  networkType: OpenShiftSDN
platform:
  openstack:
    cloud: mycloud
    externalNetwork: external
    computeFlavor: m1.xlarge
    lbFloatingIP: 128.0.0.1
fips: false
pullSecret: '{"auths": ...}'
sshKey: ssh-ed25519 AAAA...
```

9.1.10. 设置计算机器关联性

另外，您还可以在安装过程中为计算机器设置关联性策略。默认情况下，安装程序不会为计算机器选择关联性策略。

您还可以在安装后创建使用特定 RHOSP 服务器组的机器集。

注意

control plane 机器使用 **soft-anti-affinity** 策略创建。

提示

您可以在 RHOSP 文档中了解更多有关 RHOSP 实例调度和放置 的信息。

先决条件

- 创建 **install-config.yaml** 文件并完成对其所做的任何修改。

流程

1. 使用 RHOSP 命令行界面，为您的计算机器创建服务器组。例如：

```
$ openstack \
    --os-compute-api-version=2.15 \
    server group create \
    --policy anti-affinity \
    my-openshift-worker-group
```

如需更多信息，请参阅**服务器组 create** 命令文档。

2. 进入包含安装程序的目录并创建清单：

```
$ ./openshift-install create manifests --dir=<installation_directory>
```

其中：

installation_directory

指定包含集群的 **install-config.yaml** 文件的目录名称。

3. 打开 **manifests/99_openshift-cluster-api_worker-machineset-0.yaml**，这是 **MachineSet** 定义文件。

4. 将属性 **serverGroupID** 添加到 **spec.template.spec.providerSpec.value** 属性下的定义中。例如：

```
apiVersion: machine.openshift.io/v1beta1
kind: MachineSet
metadata:
  labels:
    machine.openshift.io/cluster-api-cluster: <infrastructure_ID>
    machine.openshift.io/cluster-api-machine-role: <node_role>
    machine.openshift.io/cluster-api-machine-type: <node_role>
  name: <infrastructure_ID>-<node_role>
  namespace: openshift-machine-api
spec:
  replicas: <number_of_replicas>
  selector:
    matchLabels:
      machine.openshift.io/cluster-api-cluster: <infrastructure_ID>
      machine.openshift.io/cluster-api-machineset: <infrastructure_ID>-<node_role>
  template:
    metadata:
      labels:
        machine.openshift.io/cluster-api-cluster: <infrastructure_ID>
        machine.openshift.io/cluster-api-machine-role: <node_role>
        machine.openshift.io/cluster-api-machine-type: <node_role>
        machine.openshift.io/cluster-api-machineset: <infrastructure_ID>-<node_role>
    spec:
      providerSpec:
        value:
          apiVersion: openstackproviderconfig.openshift.io/v1alpha1
          cloudName: openstack
          cloudsSecret:
            name: openstack-cloud-credentials
            namespace: openshift-machine-api
          flavor: <nova_flavor>
          image: <glance_image_name_or_location>
```

```
    serverGroupID: aaaaaaaa-bbbb-cccc-dddd-eeeeeeeeeeee 1
    kind: OpenstackProviderSpec
    networks:
    - filter: {}
      subnets:
      - filter:
          name: <subnet_name>
          tags: openshiftClusterID=<infrastructure_ID>
    securityGroups:
    - filter: {}
      name: <infrastructure_ID>-<node_role>
    serverMetadata:
      Name: <infrastructure_ID>-<node_role>
      openshiftClusterID: <infrastructure_ID>
    tags:
    - openshiftClusterID=<infrastructure_ID>
    trunk: true
    userDataSecret:
      name: <node_role>-user-data
    availabilityZone: <optional_openstack_availability_zone>
```

1 在此处添加服务器组的 UUID。

5. 可选：备份 **manifests/99_openshift-cluster-api_worker-machineset-0.yaml** 文件。创建集群时，安装程序会删除 **manifests/** 目录。

安装集群时，安装程序将使用您修改的 **MachineSet** 定义在 RHOSP 服务器组中创建计算机器。

9.1.11. 生成 SSH 私钥并将其添加到代理中

如果要在集群上执行安装调试或灾难恢复，则必须为 **ssh-agent** 和安装程序提供 SSH 密钥。您可以使用此密钥访问公共集群中的 bootstrap 机器来排除安装问题。

注意

在生产环境中，您需要进行灾难恢复和调试。

您可以使用此密钥以 **core** 用户身份通过 SSH 连接到 master 节点。在部署集群时，此密钥会添加到 **core** 用户的 **~/.ssh/authorized_keys** 列表中。

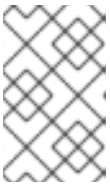
流程

1. 如果还没有为计算机上免密码身份验证而配置的 SSH 密钥，请创建一个。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ ssh-keygen -t ed25519 -N '' \
  -f <path>/<file_name> 1
```

1 指定新 SSH 密钥的路径和文件名，如 **~/.ssh/id_rsa**。如果您已有密钥对，请确保您的公钥位于 **~/.ssh** 目录中。

运行此命令会在指定的位置生成不需要密码的 SSH 密钥。



注意

如果您计划在 **x86_64** 架构中安装使用 FIPS 验证的/Modules in Process 加密库的 OpenShift Container Platform 集群，不要创建使用 **ed25519** 算法的密钥。反之，创建一个使用 **rsa** 或 **ecdsa** 算法的密钥。

2. 作为后台任务启动 **ssh-agent** 进程：

```
$ eval "$(ssh-agent -s)"
```

输出示例

```
Agent pid 31874
```

注意

如果您的集群采用 FIPS 模式，则只使用 FIPS 兼容算法来生成 SSH 密钥。密钥必须是 RSA 或 ECDSA。

3. 将 SSH 私钥添加到 **ssh-agent**：

```
$ ssh-add <path>/<file_name> (1)
```

输出示例

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

(1) 指定 SSH 私钥的路径和文件名，如 **~/.ssh/id_rsa**

后续步骤

- 在安装 OpenShift Container Platform 时，为安装程序提供 SSH 公钥。

9.1.12. 启用对环境的访问

在部署时，所有 OpenShift Container Platform 机器都是在 Red Hat OpenStack Platform (RHOSP) 租户网络中创建的。因此，大多数 RHOSP 部署中都无法直接访问它们。

您可以在安装过程中使用浮动 IP 地址（FIP）来配置 OpenShift Container Platform API 和应用程序访问。您也可以在没有配置 FIP 的情况下完成安装，但安装程序不会配置一种从外部访问 API 或应用程序的方法。

9.1.12.1. 启用通过浮动 IP 地址进行访问

创建浮动 IP（FIP）地址，用于从外部访问 OpenShift Container Platform API 和集群应用程序。

流程

1. 使用 Red Hat OpenStack Platform (RHOSP) CLI，创建 API FIP：

```
$ openstack floating ip create --description "API <cluster_name>.<base_domain>"
<external_network>
```

2. 使用 Red Hat OpenStack Platform (RHOSP) CLI，创建应用程序或 Ingress，FIP：

```
$ openstack floating ip create --description "Ingress <cluster_name>.<base_domain>"
<external_network>
```

3. 向用于 API 和 Ingress FIP 的 DNS 服务器添加符合这些模式的记录：

```
api.<cluster_name>.<base_domain>.  IN  A  <API_FIP>
*.apps.<cluster_name>.<base_domain>. IN  A <apps_FIP>
```

注意

如果您不控制 DNS 服务器，您可以通过将集群域名（如以下内容）添加到 **/etc/hosts** 文件中来访问集群：

- **<api_floating_ip> api.<cluster_name>.<base_domain>**
- **<application_floating_ip> grafana-openshift-monitoring.apps.<cluster_name>.<base_domain>**
- **<application_floating_ip> prometheus-k8s-openshift-monitoring.apps.<cluster_name>.<base_domain>**
- **<application_floating_ip> oauth-openshift.apps.<cluster_name>.<base_domain>**
- **<application_floating_ip> console-openshift-console.apps.<cluster_name>.<base_domain>**
- **application_floating_ip integrate-oauth-server-openshift-authentication.apps.<cluster_name>.<base_domain>**

/etc/hosts 文件中的集群域名授予对本地集群的 Web 控制台和监控界面的访问权限。您还可以使用 **kubectl** 或 **oc**。您可以使用指向 <application_floating_ip> 的额外条目来访问用户应用程序。此操作使 API 和应用程序可供您访问，不适合于生产部署，但允许对开发和测试进行安装。

4. 将 FIP 添加到 **install-config.yaml** 文件，将其作为以下参数的值：
   - **platform.openstack.ingressFloatingIP**
   - **platform.openstack.lbFloatingIP**

如果使用这些值，还必须在 **install-config.yaml** 文件中输入一个外部网络作为 **platform.openstack.externalNetwork** 参数的值。

提示

您可以通过分配浮动 IP 地址并更新防火墙配置，使 OpenShift Container Platform 资源在集群之外可用。

9.1.12.2. 完成没有浮动 IP 地址的安装

您可以在不提供浮动 IP 地址的情况下在 Red Hat OpenStack Platform（RHOSP）上安装 OpenShift Container Platform。

在 **install-config.yaml** 文件中，不要定义以下参数：

- **platform.openstack.ingressFloatingIP**
- **platform.openstack.lbFloatingIP**

如果您无法提供外部网络，也可以将 **platform.openstack.externalNetwork** 留空。如果没有为 **platform.openstack.externalNetwork** 提供值，则不会为您创建路由器。如果没有额外的操作，安装程序将无法从 Glance 检索镜像。您必须自行配置外部连接。

如果在因为缺少浮动 IP 地址或名称解析而无法访问集群 API 的系统中运行安装程序时，安装会失败。要防止安装失败，可以使用代理网络或者从与您的机器位于同一网络的系统中运行安装程序。

注意

您可以通过为 API 和 Ingress 端口创建 DNS 记录来启用名称解析。例如：

```
api.<cluster_name>.<base_domain>.  IN  A  <api_port_IP>
*.apps.<cluster_name>.<base_domain>. IN  A <ingress_port_IP>
```

如果您不控制 DNS 服务器，可以改为将记录添加到 **/etc/hosts** 文件中。此操作使 API 可供您自己访问，不适合于生产部署。这可用于进行开发和测试的安装。

9.1.13. 部署集群

您可以在兼容云平台中安装 OpenShift Container Platform。

重要

安装程序的 **create cluster** 命令只能在初始安装过程中运行一次。

先决条件

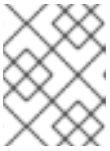
- 获取 OpenShift Container Platform 安装程序以及集群的 pull secret。

流程

1. 更改为包含安装程序的目录并初始化集群部署：

```
$ ./openshift-install create cluster --dir <installation_directory> \ 1
    --log-level=info 2
```

1. 对于 **<installation_directory>**，请指定自定义 **./install-config.yaml** 文件的位置。
2. 要查看不同的安装详情，请指定 **warn**、**debug** 或 **error**，而不要指定 **info**。



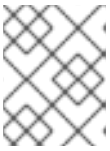
注意

如果您在主机上配置的云供应商帐户没有足够的权限来部署集群，安装过程将会停止，并且显示缺少的权限。

集群部署完成后，终端会显示访问集群的信息，包括指向其 Web 控制台的链接和 **kubeadmin** 用户的凭证。

输出示例

```
...
INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run 'export
KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'
INFO Access the OpenShift web-console here: https://console-openshift-
console.apps.mycluster.example.com
INFO Login to the console with user: "kubeadmin", and password: "4vYBz-Ee6gm-ymBZj-
Wt5AL"
INFO Time elapsed: 36m22s
```



注意

当安装成功时，集群访问和凭证信息还会输出到 **<installation_directory>/.openshift_install.log**。

重要

- 安装程序生成的 Ignition 配置文件包含在 24 小时后过期的证书，然后在过期时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外情况是，您需要手动批准待处理的 **node-bootstrapper** 证书签名请求（CSR）来恢复 kubelet 证书。如需更多信息，请参阅*从过期的 control plane 证书中恢复*的文档。
- 建议您在生成 12 小时后使用 Ignition 配置文件，因为集群安装后 24 小时证书从 16 小时轮转至 22 小时。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中运行证书更新时避免安装失败。

重要

您不得删除安装程序或安装程序所创建的文件。需要这两者才能删除集群。

9.1.14. 验证集群状态

您可以在安装过程中或安装后验证 OpenShift Container Platform 集群的状态：

流程

1. 在集群环境中，导出管理员的 kubeconfig 文件：

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

kubeconfig 文件包含关于集群的信息，供 CLI 用于将客户端连接到正确集群和 API 服务器。

2. 查看部署后创建的 control plane 和计算机器：

```
$ oc get nodes
```

3. 查看集群的版本：

```
$ oc get clusterversion
```

4. 查看 Operator 的状态：

```
$ oc get clusteroperator
```

5. 查看集群中的所有正在运行的 pod:

```
$ oc get pods -A
```

9.1.15. 使用 CLI 登录到集群

您可以通过导出集群 **kubeconfig** 文件，以默认系统用户身份登录集群。**kubeconfig** 文件包含关于集群的信息，供 CLI 用于将客户端连接到正确集群和 API 服务器。该文件特只适用于一个特定的集群，在 OpenShift Container Platform 安装过程中创建。

先决条件

- 已部署了 OpenShift Container Platform 集群。
- 已安装 **oc** CLI。

流程

1. 导出 **kubeadmin** 凭证：

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

2. 使用导出的配置，验证能否成功运行 **oc** 命令：

```
$ oc whoami
```

输出示例

```
system:admin
```

其他资源

- 如需有关访问和了解 OpenShift Container Platform Web 控制台的更多信息，请参阅访问 Web 控制台。

9.1.16. OpenShift Container Platform 的 Telemetry 访问

在 OpenShift Container Platform 4.6 中，默认运行的 Telemetry 服务提供有关集群健康状况和成功更新的指标，需要访问互联网。如果您的集群连接到互联网，Telemetry 会自动运行，而且集群会注册到 OpenShift Cluster Manager。

确认 OpenShift Cluster Manager 清单正确后，可以由 Telemetry 自动维护，也可以使用 OpenShift Cluster Manager 手动维护，使用订阅监控来跟踪帐户或多集群级别的 OpenShift Container Platform 订阅。

其他资源

- 有关 Telemetry 服务的更多信息，请参阅关于远程健康监控。

9.1.17. 后续步骤

- 自定义集群。
- 如果需要，您可以选择不使用远程健康报告。
- 如果您需要启用对节点端口的外部访问，请使用节点端口配置集群流量。
- 如果您没有将 RHOSP 配置为使用浮动 IP 地址接受应用程序流量，使用浮动 IP 地址配置 RHOSP 访问。

9.2. 在带有 KURYR 的 OPENSTACK 上安装集群

在 OpenShift Container Platform 版本 4.6 中，您可以在使用 Kuryr SDN 的 Red Hat OpenStack Platform（RHOSP）上安装自定义集群。要自定义安装，请在安装集群前修改 **install-config.yaml** 中的参数。

9.2.1. 先决条件

- 查看有关 OpenShift Container Platform 安装和更新流程的详细信息。
  - 在 *Available platforms* 部分验证 OpenShift Container Platform 4.6 是否与您的 RHOSP 版本兼容。您还可以查看 OpenShift Container Platform 在 RHOSP 中的支持 来比较不同版本的平台支持。
- 验证您的网络配置不依赖于供应商网络。不支持提供商网络。
- 在 RHOSP 中安装了存储服务，如快存储（Cinder）或对象存储（Swift）。对象存储是 OpenShift Container Platform registry 集群部署的推荐存储技术。如需更多信息，请参阅 优化存储。

9.2.2. 关于 Kuryr SDN

Kuryr 是一个容器网络接口（CNI）插件解决方案，它使用 Neutron 和 Octavia Red Hat OpenStack Platform（RHOSP）服务来为 pod 和服务提供网络。

Kuryr 和 OpenShift Container Platform 的集成主要针对在 RHOSP VM 上运行的 OpenShift Container Platform 集群设计。Kuryr 通过将 OpenShift Container Platform pod 插入到 RHOSP SDN 来提高网络性能。另外，它还提供 pod 和 RHOSP 虚拟实例间的互联性。

Kuryr 组件作为 pod 在 OpenShift Container Platform 中安装，使用 **openshift-kuryr** 命名空间：

- **kuryr-controller** - 在一个 **master** 节点上安装的单个服务实例。这在 OpenShift Container Platform 中建模为一个 **Deployment** 对象。
- **kuryr-cni** - 在每个 OpenShift Container Platform 节点上安装并配置 Kuryr 作为 CNI 驱动的容器。这在 OpenShift Container Platform 中建模为一个 **DaemonSet** 对象。

Kuryr 控制器监控 OpenShift Container Platform API 服务器中的 pod、服务和命名空间创建、更新和删除事件。它将 OpenShift Container Platform API 调用映射到 Neutron 和 Octavia 中的对应对象。这意味着，实现了 Neutron 中继端口功能的每个网络解决方案都可以通过 Kuryr 支持 OpenShift Container Platform。这包括开源解决方案，比如 Open vSwitch (OVS) 和 Open Virtual Network (OVN)，以及 Neutron 兼容的商业 SDN。

建议在封装的 RHOSP 租户网络上部署 OpenShift Container Platform 时使用 Kuryr，以避免出现重复封装，例如通过 RHOSP 网络运行封装的 OpenShift Container Platform SDN。

如果您使用供应商网络或租户 VLAN，则不需要使用 Kuryr 来避免重复封装。虽然性能上的优势微不足道，但根据您的配置，使用 Kuryr 避免两个覆盖可能仍然有用。

在完足以下所有条件的部署中不建议使用 Kuryr：

- RHOSP 版本早于 16
- 部署使用 UDP 服务，或者在几个 hypervisor 上使用大量 TCP 服务。

或

- **ovn-octavia** Octavia 驱动被禁用。
- 部署在几个 hypervisor 中使用了大量的 TCP 服务。

9.2.3. 在带有 Kuryr 的 OpenStack 上安装 OpenShift Container Platform 的资源指南

当使用 Kuryr SDN 时，pod、服务、命名空间和网络策略会使用来自 RHOSP 配额的资源，这会增加最低要求。除了默认安装需要满足的要求，Kuryr 还有一些额外的要求。

使用以下配额来满足集群的默认最低要求：

表 9.7. 带有 Kuryr 的 RHOSP 上默认 OpenShift Container Platform 集群的建议资源

资源	值
浮动 IP 地址	3 - 加上预期的 LoadBalancer 类型服务的数量
端口	1500 - 每个 Pod 需要 1 个
路由器	1
子网	250 - 每个命名空间/项目需要 1 个
网络	250 - 每个命名空间/项目需要 1 个
RAM	112 GB

资源	值
vCPUs	28
卷存储	275 GB
实例	7
安全组	250 - 每个服务和每个 NetworkPolicy 需要 1 个
安全组规则	1000
负载均衡器	100 - 每个服务需要 1 个
负载均衡器侦听程序	500 - 每个服务公开端口需要 1 个
负载均衡器池	500 - 每个服务公开端口需要 1 个

集群或许能使用少于推荐数量的资源来运作，但其性能无法保证。

重要

如果 RHOSP 对象存储（Swift）可用，并由具有 **swiftoperator** 角色的用户帐户执行，它会作为 OpenShift Container Platform 镜像 registry 的默认后端。在这种情况下，卷存储需要有 175GB。根据镜像 registry 的大小，Swift 空间要求会有所不同。

重要

如果您使用带有 Amphora 驱动而不是 OVN Octavia 驱动的 Red Hat OpenStack Platform (RHOSP) 版本 16，则安全组会与服务帐户而不是用户项目关联。

在设置资源时请考虑以下几点：

- 需要的端口数量会大于 pod 的数量。Kuryr 使用端口池来预创建端口以供 pod 使用，用于加快 pod 的启动时间。
- 每个网络策略都映射到 RHOSP 安全组中，并根据 **NetworkPolicy** 规格将一个或多个规则添加到安全组中。
- 每个服务都映射到一个 RHOSP 负载均衡器中。在估算配额所需安全组数时，请考虑此要求。
  如果您使用 RHOSP 版本 15 或更早版本，或者使用 **ovn-octavia** 驱动，则每个负载均衡器都有一个带有用户项目的安全组。
- 配额不考虑负载均衡器资源（如 VM 资源），但您必须在决定 RHOSP 部署的大小时考虑这些资源。默认安装将有超过 50 个负载均衡器，集群必须可以容纳它们。
  如果您使用启用 OVN Octavia 驱动程序的 RHOSP 版本 16，则只生成一个负载均衡器虚拟机; 服务通过 OVN 流平衡负载。

OpenShift Container Platform 部署由 control plane 机器、计算机器和 bootstrap 机器组成。

要启用 Kuryr SDN，您的环境必须满足以下要求：

- 运行 RHOSP 13+。
- 具有 Octavia 的 Overcloud。
- 使用 Neutron Trunk 端口扩展。
- 如果使用 ML2/OVS Neutron 驱动而不是 **ovs-hybrid**，则请使用 **openvswitch** 防火墙驱动。

9.2.3.1. 增加配额

使用 Kuryr SDN 时，您必须提高配额以满足 pod 、Services 、namespaces 和网络策略所使用的 Red Hat OpenStack Platform（RHOSP）资源要求。

流程

- 运行以下命令为项目增加配额：

```
$ sudo openstack quota set --secgroups 250 --secgroup-rules 1000 --ports 1500 --subnets
250 --networks 250 <project>
```

9.2.3.2. 配置 Neutron

Kuryr CNI 利用 Neutron Trunks 扩展来将容器插入 Red Hat OpenStack Platform（RHOSP）SDN，因此您必须使用 **trunks** 扩展才可以使 Kuryr 正常工作。

另外，如果您使用默认的 ML2/OVS Neutron 驱动程序，防火墙必须设为 **openvswitch** 而不是 **ovs_hybrid** ，以便在中继子端口上强制实施安全组，同时 Kuryr 可以正确处理网络策略。

9.2.3.3. 配置 Octavia

Kuryr SDN 使用 Red Hat OpenStack Platform（RHOSP）的 Octavia LBaaS 来实现 OpenShift Container Platform 服务。因此，您必须在 RHOSP 上安装和配置 Octavia 组件以使用 Kuryr SDN。

要启用 Octavia，您必须在安装 RHOSP Overcloud 的过程中包括 Octavia 服务，如果 Overcloud 已存在则需要升级 Octavia 服务。以下启用 Octavia 的步骤适用于新的 Overcloud 安装或 Overcloud 更新。

注意

以下步骤只包括在部署 RHOSP 时需要处理 Octavia 部分的信息。请注意 registry 可能会不同。

这个示例使用本地的 registry。

流程

1. 如果您使用本地 registry，请创建一个模板来将镜像上传到 registry。例如：

```
(undercloud) $ openstack overcloud container image prepare \
-e /usr/share/openstack-tripleo-heat-templates/environments/services-docker/octavia.yaml \
--namespace=registry.access.redhat.com/rhosp13 \
--push-destination=<local-ip-from-undercloud.conf>:8787 \
--prefix=openstack- \
```

```
--tag-from-label {version}-{release} \
--output-env-file=/home/stack/templates/overcloud_images.yaml \
--output-images-file /home/stack/local_registry_images.yaml
```

2. 验证 **local_registry_images.yaml** 文件是否包含 Octavia 镜像。例如：

```
...
- imagename: registry.access.redhat.com/rhosp13/openstack-octavia-api:13.0-43
  push_destination: <local-ip-from-undercloud.conf>:8787
- imagename: registry.access.redhat.com/rhosp13/openstack-octavia-health-manager:13.0-
45
  push_destination: <local-ip-from-undercloud.conf>:8787
- imagename: registry.access.redhat.com/rhosp13/openstack-octavia-housekeeping:13.0-45
  push_destination: <local-ip-from-undercloud.conf>:8787
- imagename: registry.access.redhat.com/rhosp13/openstack-octavia-worker:13.0-44
  push_destination: <local-ip-from-undercloud.conf>:8787
```

注意

Octavia 容器版本根据所安装的特定 RHOSP 版本的不同而有所不同。

3. 将 **registry.redhat.io** 中的容器镜像拉取到 Undercloud 节点：

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

这可能需要一些时间，具体要看您的网络速度和 Undercloud 使用的磁盘。

4. 由于 Octavia 负载均衡器是用来访问 OpenShift Container Platform API，所以您必须增加它们的监听程序的默认超时时间。默认超时为 50 秒。通过将以下文件传递给 Overcloud deploy 命令，将超时时间增加到 20 分钟：

```
(undercloud) $ cat octavia_timeouts.yaml
parameter_defaults:
  OctaviaTimeoutClientData: 1200000
  OctaviaTimeoutMemberData: 1200000
```

注意

RHOSP 13.0.13+ 不需要这一步。

5. 使用 Octavia 安装或更新 overcloud 环境：

```
$ openstack overcloud deploy --templates \
  -e /usr/share/openstack-tripleo-heat-templates/environments/services-docker/octavia.yaml \
  -e octavia_timeouts.yaml
```

注意

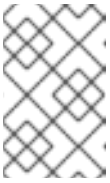
这个命令只包含与 Octavia 相关的文件，它根据您具体的 RHOSP 安装而有所不同。如需更多信息，请参阅 RHOSP 文档。有关自定义 Octavia 安装的详情请参考使用 Director 安装 Octavia。

注意

当利用 Kuryr SDN 时，Overcloud 安装需要 Neutron **trunk** 扩展。这在 director 部署中默认可用。当 Neutron 后端是 ML2/OVS 时，使用 **openvswitch** 防火墙而不是默认的 **ovs-hybrid**。如果后端为 ML2/OVN，则不需要修改。

6. 在早于 13.0.13 的 RHOSP 版本中，在创建项目后将项目 ID 添加到 **octavia.conf** 配置文件中。

   - 要跨服务实施网络策略，比如网络流量会通过 Octavia 负载均衡器时，您必须确保 Octavia 在用户项目中创建 Amphora VM 安全组。
     这可确保所需的 LoadBalancer 安全组属于该项目，并可将其更新为强制实施服务隔离。

     注意

     在 RHOSP 13.0.13 或更高版本中不需要此操作。

     Octavia 实施新的 ACL API，限制对负载均衡器 VIP 的访问。

     a. 获取项目 ID

     ```
     $ openstack project show <project>
     ```

     输出示例

     ```
     +-------------+----------------------------------+
     | Field       | Value                            |
     +-------------+----------------------------------+
     | description |                                  |
     | domain_id   | default                          |
     | enabled     | True                             |
     | id          | PROJECT_ID                       |
     | is_domain   | False                            |
     | name        | *<project>*                      |
     | parent_id   | default                          |
     | tags        | []                               |
     +-------------+----------------------------------+
     ```

     b. 将项目 ID 添加到控制器的 **octavia.conf** 中。

        i. Source **stackrc** 文件：

        ```
        $ source stackrc  # Undercloud credentials
        ```

        ii. 列出 Overcloud 控制器。

        ```
        $ openstack server list
        ```

输出示例

```
+--------------------------------------+-------------+--------+----------------------+-------
--------+------------+
|
| ID                                   | Name        | Status | Networks
| Image         | Flavor     |
|
+--------------------------------------+-------------+--------+----------------------+-------
--------+------------+
|
| 6bef8e73-2ba5-4860-a0b1-3937f8ca7e01 | controller-0 | ACTIVE |
ctlplane=192.168.24.8 | overcloud-full | controller |
|
| dda3173a-ab26-47f8-a2dc-8473b4a67ab9 | compute-0    | ACTIVE |
ctlplane=192.168.24.6 | overcloud-full | compute    |
|
+--------------------------------------+-------------+--------+----------------------+-------
--------+------------+
```

iii. SSH 到控制器。

```
$ ssh heat-admin@192.168.24.8
```

iv. 编辑 **octavia.conf** 文件，将项目添加到 Amphora 安全组存在于用户账户的项目列表中。

```
# List of project IDs that are allowed to have Load balancer security groups
# belonging to them.
amp_secgroup_allowed_projects = PROJECT_ID
```

c. 重启 Octavia worker 以便重新加载配置。

```
controller-0$ sudo docker restart octavia_worker
```

注意

根据您的 RHOSP 环境，Octavia 可能不支持 UDP 侦听程序。如果您在 RHOSP 版本 13.0.13 或更早版本使用 Kuryr SDN，则不支持 UDP 服务。RHOSP 版本 16 或更高版本支持 UDP。

9.2.3.3.1. Octavia OVN 驱动程序

Octavia 通过 Octavia API 支持多个供应商驱动程序。

要查看所有可用的 Octavia 提供程序驱动，请在命令行中输入：

```
$ openstack loadbalancer provider list
```

输出示例

```
+---------+-------------------------------------------------+
| name    | description                                     |
```

```
+---------+-----------------------------------------------+
| amphora | The Octavia Amphora driver.                   |
| octavia | Deprecated alias of the Octavia Amphora driver. |
| ovn     | Octavia OVN driver.                           |
+---------+-----------------------------------------------+
```

从 RHOSP 版本 16 开始，Octavia OVN 供应商驱动程序（**ovn**）在 RHOSP 部署的 OpenShift Container Platform 上被支持。

ovn 是 Octavia 和 OVN 提供的负载平衡集成驱动。它支持基本负载平衡功能，并基于 OpenFlow 规则。在使用 OVN Neutron ML2 的部署中，Director 会在 Octavia 中自动启用该驱动程序。

Amphora 供应商驱动程序是默认驱动程序。如果启用了 **ovn**，Kuryr 将使用它。

如果 Kuryr 使用 **ovn** 而不是 Amphora，则可提供以下优点：

- 资源要求更低Kuryr 不需要为每个服务都提供一个负载均衡器虚拟机。
- 网络延迟会降低。
- 通过对每个服务使用 OpenFlow 规则而不是 VM 来提高服务创建速度。
- 跨所有节点的分布式负载平衡操作，而不是集中到 Amphora 虚拟机中。

您可以在 RHOSP 云从版本 13 升级到版本 16 后，将集群配置为使用 Octavia OVN 驱动程序。

9.2.3.4. 已知使用 Kuryr 安装的限制

将 OpenShift Container Platform 与 Kuryr SDN 搭配使用有一些已知的限制。

RHOSP 常规限制
带有 Kuryr SDN 的 OpenShift Container Platform 不支持带有类型 **NodePort** 的 **Service** 对象。

如果机器子网没有连接到路由器，或者子网已连接，但路由器没有设置外部网关，Kuryr 无法为类型为 **LoadBalancer** 的 **Service** 对象创建浮动 IP。

- 在 **Service** 对象上配置 **sessionAffinity=ClientIP** 属性无效。Kuryr 不支持此设置。

RHOSP 版本限制
使用带有 Kuryr SDN 的 OpenShift Container Platform 有一些限制，具体取决于 RHOSP 版本。

- RHOSP 16 之前的版本使用默认 Octavia 负载均衡器驱动程序(Amphora)。此驱动要求在每个 OpenShift Container Platform 服务中部署一个 Amphora 负载均衡器虚拟机。创建太多的服务会导致您耗尽资源。
  如果以后版本的 RHOSP 部署中禁用了 OVN Octavia 驱动程序，则也会使用 Amphora 驱动。它们对资源的要求和早期版本 RHOSP 相同。
- Octavia RHOSP 13.0.13 之前的版本不支持 UDP 侦听程序。因此，OpenShift Container Platform UDP 服务不被支持。
- Octavia RHOSP 13.0.13 之前的版本无法侦听同一端口上的多个协议。不支持将同一端口暴露给不同协议的服务，比如 TCP 和 UDP。
- Kuryr SDN 不支持由服务自动取消闲置。

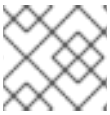
RHOSP 环境限制
使用取决于您的部署环境的 Kuryr SDN 会有一些限制。

由于 Octavia 缺少对 UDP 协议和多个监听器的支持，如果 rhosp 版本早于 13.0.13，Kuryr 会强制 pod 在 DNS 解析中使用 TCP，如果：

在 Go 版本 1.12 及更早的版本中，通过 CGO 支持被禁用的模式编译的应用程序只使用 UDP。在这种情况下，native Go 解析器无法识别 **resolv.conf** 中的 **use-vc** 选项，它控制 DNS 解析是否强制使用 TCP。因此，UDP 仍会被用来解析 DNS，这将导致失败。

要确保 TCP 强制使用是允许的，在编译应用程序使把环境变量 **CGO_ENABLED** 设定为 **1**（如 **CGO_ENABLED=1**），或者不使用这个变量。

在 Go 版本 1.13 及之后的版本中，如果使用 UDP 的 DNS 解析失败，则会自动使用 TCP。



注意

基于 musl 的容器，包括基于 Alpine 的容器，不支持 **use-vc** 选项。

RHOSP 升级限制
作为 RHOSP 升级过程的结果，可能会更改 Octavia API，并可能需要升级到用于负载均衡器的 Amphora 镜像。

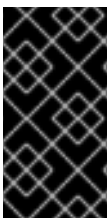
您可以单独处理 API 更改。

如果升级了 Amphora 镜像，RHOSP Operator 可使用两种方式处理现有的负载均衡器虚拟机：

- 通过触发负载均衡器故障切换来升级每个虚拟机。
- 将升级虚拟机的职责留给用户。

如果运算符使用第一个选项，在故障切换过程中可能会有短暂的停机时间。

如果 Operator 采用第二个选项，现有负载均衡器将不支持升级的 Octavia API 功能，比如 UDP 侦听程序。在这种情况下，用户必须重新创建自己的服务以使用这些功能。



重要

如果 OpenShift Container Platform 检测到支持 UDP 负载均衡的新 Octavia 版本，它会自动重新创建 DNS 服务。服务重新创建可确保服务默认支持 UDP 负载均衡。

这个重新创建会导致 DNS 服务大约停机 一 分钟。

9.2.3.5. control plane 机器

默认情况下，OpenShift Container Platform 安装过程会创建三台 control plane 机器。

每台机器都需要：

- 来自 RHOSP 配额的实例
- 来自 RHOSP 配额的端口
- 至少 16 GB 内存、4 个 vCPU 和 100 GB 存储空间的类别

9.2.3.6. 计算机器

默认情况下，OpenShift Container Platform 安装过程会创建三台计算机器。

每台机器都需要：

- 来自 RHOSP 配额的实例
- 来自 RHOSP 配额的端口
- 至少有 8 GB 内存、2 个 vCPU 和 100 GB 存储空间的类别

提示

计算机器托管您在 OpenShift Container Platform 上运行的应用程序；运行数量应尽可能多。

9.2.3.7. bootstrap 机器

在安装时，会临时置备 bootstrap 机器来支持 control plane。生产控制平面就绪后，bootstrap 机器会被取消置备。

bootstrap 机器需要：

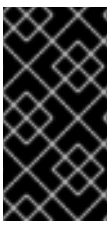
- 来自 RHOSP 配额的实例
- 来自 RHOSP 配额的端口
- 至少 16 GB 内存、4 个 vCPU 和 100 GB 存储空间的类别

9.2.4. OpenShift Container Platform 的互联网访问

在 OpenShift Container Platform 4.6 中，您需要访问互联网来安装集群。

您必须具有以下互联网访问权限：

- 访问 OpenShift Cluster Manager 以下载安装程序并执行订阅管理。如果集群可以访问互联网，并且没有禁用 Telemetry，该服务会自动授权您的集群。
- 访问 Quay.io，以获取安装集群所需的软件包。
- 获取执行集群更新所需的软件包。

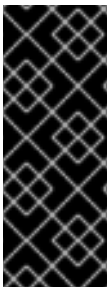


重要

如果您的集群无法直接访问互联网，则可以在置备的某些类基础架构上执行受限网络安装。在此过程中，您要下载所需的内容，并使用它在镜像 registry（mirror registry） 中填充安装集群并生成安装程序所需的软件包。对于某些安装类型，集群要安装到的环境不需要访问互联网。在更新集群之前，要更新 registry 镜像系统中的内容。

9.2.5. 在 RHOSP 上启用 Swift

Swift 由具有 **swiftoperator** 角色的用户帐户操控。在运行安装程序前，将该角色添加到帐户。



重要

如果 Red Hat OpenStack Platform（RHOSP） 对象存储服务 （通常称为 Swift）可用，OpenShift Container Platform 会使用它作为镜像 registry 存储。如果无法使用，安装程序将依赖于 RHOSP 快存储服务，通常称为 Cinder 。

如果 Swift 存在且您想要使用 Swift，则必须启用对其的访问。如果不存在，或者您不想使用它，请跳过这个部分。

先决条件

- 在目标环境中具有 RHOSP 管理员帐户
- 已安装 Swift 服务。
- 在Ceph RGW 上启用了**account in url** 选项。

流程

在 RHOSP 上启用 Swift：

1. 在 RHOSP CLI 中以管理员身份，将 **swiftoperator** 角色添加到要访问 Swift 的帐户：

```
$ openstack role add --user <user> --project <project> swiftoperator
```

您的 RHOSP 部署现可以使用 Swift 用于镜像 registry。

9.2.6. 验证外部网络访问

OpenShift Container Platform 安装进程需要外部网络访问权限。您必须为其提供外部网络值，否则部署会失败。在运行安装进程前，请验证 Red Hat OpenStack Platform (RHOSP) 中是否存在具有外部路由器类型的网络。

先决条件

- 将 OpenStack 联网服务配置为使用 DHCP 代理转发实例 DNS 查询

流程

1. 使用 RHOSP CLI 验证"外部"网络的名称和 ID：

```
$ openstack network list --long -c ID -c Name -c "Router Type"
```

输出示例

```
+--------------------------------------+----------------+-------------+
| ID                                   | Name           | Router Type |
+--------------------------------------+----------------+-------------+
| 148a8023-62a7-4672-b018-003462f8d7dc | public_network | External    |
+--------------------------------------+----------------+-------------+
```

网络列表中会显示具有外部路由器类型的网络。如果最少有一个没有，请参阅 创建默认浮动 IP 网络和创建默认供应商网络。

重要

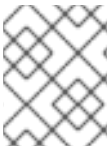
如果外部网络 CIDR 范围与某一个默认网络范围重叠，您必须在运行安装进程前更改 **install-config.yaml** 文件中匹配的网络范围。

默认的网络范围：

网络	范围
machineNetwork	10.0.0.0/16
serviceNetwork	172.30.0.0/16
clusterNetwork	10.128.0.0/14

警告

如果安装程序找到多个同名的镜像，它会随机设置其中之一。为避免这种行为，请在 RHOSP 中为资源创建唯一名称。



注意

如果启用了 Neutron 中继服务插件，则默认创建中继端口。如需更多信息，请参阅 Neutron 中继端口。

9.2.7. 为安装程序定义参数

OpenShift Container Platform 安装程序依赖于一个名为 **clouds.yaml** 的文件。该文件描述了 Red Hat OpenStack Platform (RHOSP) 配置参数，包括项目名称、登录信息和授权服务 URL。

流程

1. 创建 **clouds.yaml** 文件：
   - 如果您的 RHOSP 发行版包含 Horizon web UI，请在该 UI 中生成 **clouds.yaml** 文件。

     重要

     请记住在 **auth** 字段中添加密码。您也可以把 secret 保存在 **clouds.yaml** 以外的一个独立的文件中。

   - 如果您的 RHOSP 发行版不包含 Horizon Web UI，或者您不想使用 Horizon，请自行创建该文件。如需有关 **clouds.yaml** 的详细信息，请参阅 RHOSP 文档中的配置文件。

     ```
     clouds:
       shiftstack:
         auth:
     ```

```
    auth_url: http://10.10.14.42:5000/v3
    project_name: shiftstack
    username: shiftstack_user
    password: XXX
    user_domain_name: Default
    project_domain_name: Default
dev-env:
  region_name: RegionOne
  auth:
    username: 'devuser'
    password: XXX
    project_name: 'devonly'
    auth_url: 'https://10.10.14.22:5001/v2.0'
```

2. 如果您的 RHOSP 安装使用自签名证书颁发机构 (CA) 证书进行端点身份验证：

   a. 将 CA 文件复制到您的机器中。

   b. 将机器添加到证书颁发机构信任捆绑包中：

   ```
   $ sudo cp ca.crt.pem /etc/pki/ca-trust/source/anchors/
   ```

   c. 更新信任捆绑包：

   ```
   $ sudo update-ca-trust extract
   ```

   d. 将 **cacerts** 键添加到 **clouds.yaml** 文件。该值必须是到 CA 证书的绝对路径，则其可以被非根用户访问：

   ```
   clouds:
     shiftstack:
       ...
       cacert: "/etc/pki/ca-trust/source/anchors/ca.crt.pem"
   ```

   提示

   使用自定义 CA 证书运行安装程序后，您可以通过编辑 **cloud-provider-config** keymap 中的**ca-cert.pem** 键的值来更新证书。在命令行中运行：

   ```
   $ oc edit configmap -n openshift-config cloud-provider-config
   ```

3. 将 **clouds.yaml** 文件放在以下位置之一：

   a. **OS_CLIENT_CONFIG_FILE** 环境变量的值

   b. 当前目录

   c. 特定于 Unix 的用户配置目录，如 **~/.config/openstack/clouds.yaml**

   d. 特定于 Unix 的站点配置目录，如 **/etc/openstack/clouds.yaml**
   安装程序会按照以上顺序搜索 **clouds.yaml**。

9.2.8. 获取安装程序

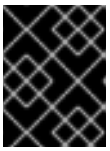
在安装 OpenShift Container Platform 之前，将安装文件下载到本地计算机上。

先决条件

- 运行 Linux 或 macOS 的计算机，本地磁盘空间为 500 MB

流程

1. 访问 OpenShift Cluster Manager 站点的 Infrastructure Provider 页面。如果您有红帽帐号，请使用自己的凭证登录。如果没有，请创建一个帐户。
2. 选择您的基础架构供应商。
3. 进入适用于您的安装类型的页面，下载您的操作系统的安装程序，并将文件放在要保存安装配置文件的目录中。。



重要

安装程序会在用来安装集群的计算机上创建若干文件。在完成集群安装后，您必须保留安装程序和安装程序所创建的文件。这两个文件都需要删除集群。

重要

删除安装程序创建的文件不会删除您的集群，即使集群在安装过程中失败也是如此。要删除集群，为特定云供应商完成 OpenShift Container Platform 卸载流程。

4. 提取安装程序。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ tar xvf openshift-install-linux.tar.gz
```

5. 从 Red Hat OpenShift Cluster Manager 下载安装 pull secret 。通过此 pull secret，您可以进行所含授权机构提供的服务的身份验证，这些服务包括为 OpenShift Container Platform 组件提供容器镜像的 Quay.io。

9.2.9. 创建安装配置文件

您可以自定义在 Red Hat OpenStack Platform（RHOSP）上安装的 OpenShift Container Platform 集群。

先决条件

- 获取 OpenShift Container Platform 安装程序以及集群的 pull secret。

流程

1. 创建 **install-config.yaml** 文件。
   a. 更改到包含安装程序的目录，再运行以下命令：

```
$ ./openshift-install create install-config --dir <installation_directory> 1
```

1 对于 **<installation_directory>**，请指定用于保存安装程序所创建的文件的目录名称。

重要

指定一个空目录。一些安装信息，如 bootstrap X.509 证书，有较短的过期间隔，因此不要重复使用安装目录。如果要重复使用另一个集群安装中的个别文件，可以将其复制到您的目录中。但是，一些安装数据的文件名可能会在发行版本之间有所改变。从 OpenShift Container Platform 老版本中复制安装文件时要格外小心。

b. 在提示符处，提供您的云的配置详情：

i. 可选：选择用来访问集群机器的 SSH 密钥。

注意

对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 **ssh-agent** 进程使用的 SSH 密钥。

ii. 选择 **openstack** 作为目标平台。

iii. 指定用于安装集群的 Red Hat OpenStack Platform (RHOSP) 外部网络名称。

iv. 指定用于从外部访问 OpenShift API 的浮动 IP 地址。

v. 指定至少有 16 GB RAM 用于 control plane 节点，以及计算节点的 8 GB RAM。

vi. 选择集群要部署到的基域。所有 DNS 记录都将是这个基域的子域，并包含集群名称。

vii. 为集群输入一个名称。名称不能多于 14 个字符。

viii. 粘贴 Red Hat OpenShift Cluster Manager 中的 pull secret 。

2. 修改 **install-config.yaml** 文件。您可以在**安装配置参数**部分中找到有关可用参数的更多信息。

3. 备份 **install-config.yaml** 文件，以便用于安装多个集群。

重要

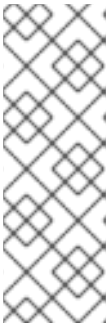
install-config.yaml 文件会在安装过程中消耗掉。如果要重复使用此文件，必须现在备份。

9.2.9.1. 在安装过程中配置集群范围代理

生产环境可能会拒绝直接访问互联网，而是提供 HTTP 或 HTTPS 代理。您可以通过在 **install-config.yaml** 文件中配置代理设置，将新的 OpenShift Container Platform 集群配置为使用代理。

先决条件

- 您有一个现有的 **install-config.yaml** 文件。
- 您检查了集群需要访问的站点，并决定是否需要绕过代理。默认情况下代理所有集群出口流量，包括对托管云供应商 API 的调用。您需要将站点添加到 **Proxy** 对象的 **spec.noProxy** 字段来绕过代理。



注意

Proxy 对象 **status.noProxy** 字段使用安装配置中的 **networking.machineNetwork[].cidr**、**networking.clusterNetwork[].cidr** 和 **networking.serviceNetwork[]** 字段的值填充。

对于在 Amazon Web Services(AWS)、Google Cloud Platform(GCP)、Microsoft Azure 和 Red Hat OpenStack Platform(RHOSP)上安装, **Proxy** 对象 **status.noProxy** 字段也会使用实例元数据端点填充(**169.254.169.254**)。

流程

1. 编辑 **install-config.yaml** 文件并添加代理设置。例如：

```
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: example.com 3
additionalTrustBundle: | 4
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
...
```

1. 用于创建集群外 HTTP 连接的代理 URL。URL 必须是 **http**。
2. 用于创建集群外 HTTPS 连接的代理 URL。
3. 要排除在代理中的目标域名、IP 地址或其他网络 CIDR 的逗号分隔列表。在域前面加 **.** 来仅匹配子域。例如： **.y.com** 匹配 **x.y.com**，但不匹配 **y.com**。使用 * 绕过所有目的地的代理。
4. 如果提供，安装程序会在 **openshift-config 命名空间中生成名为 user-ca- bundle** 的配置映射来保存额外的 CA 证书。如果您提供 **additionalTrustBundle** 和至少一个代理设置，则 **Proxy** 对象会被配置为引用 **trustedCA** 字段中的 **user-ca-bundle** 配置映射。然后，Cluster Network Operator 会创建一个 **trusted-ca-bundle** 配置映射，该配置映射将为 **trustedCA** 参数指定的内容与 RHCOS 信任捆绑包合并。 **additionalTrustBundle** 字段是必需的，除非代理的身份证书由来自 RHCOS 信任捆绑包的颁发机构签名。

注意

安装程序不支持代理的 **readinessEndpoints** 字段。

2. 保存该文件，并在安装 OpenShift Container Platform 时引用。

安装程序会创建一个名为 **cluster** 的集群范围代理，该代理使用提供的 **install-config.yaml** 文件中的代理设置。如果没有提供代理设置，仍然会创建一个 **cluster Proxy** 对象，但它会有一个空 **spec**。

注意

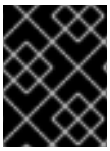
只支持名为 **cluster** 的 **Proxy** 对象，且无法创建额外的代理。

9.2.10. 安装配置参数

在部署 OpenShift Container Platform 集群前，您可以提供参数值，以描述托管集群的云平台的帐户并选择性地自定义集群平台。在创建 **install-config.yaml** 安装配置文件时，您可以通过命令行来提供所需的参数的值。如果要自定义集群，可以修改 **install-config.yaml** 文件来提供关于平台的更多信息。

注意

安装之后，您无法修改 **install-config.yaml** 文件中的这些参数。



重要

openshift-install 命令不验证参数的字段名称。如果指定了不正确的名称，则不会创建相关的文件或对象，且不会报告错误。确保所有指定的参数的字段名称都正确。

9.2.10.1. 所需的配置参数

下表描述了所需的安装配置参数：

表 9.8. 所需的参数

参数	描述	值
apiVersion	**install-config.yaml** 内容的 API 版本。当前版本是 **v1**。安装程序还可能支持旧的 API 版本。	字符串
baseDomain	云供应商的基域。此基础域用于创建到 OpenShift Container Platform 集群组件的路由。集群的完整 DNS 名称是 **baseDomain** 和 **metadata.name** 参数值的组合，其格式为 **<metadata.name>.<baseDomain>**。	完全限定域名或子域名，如 **example.com**。
metadata	Kubernetes 资源 **ObjectMeta**，其中只消耗 **name** 参数。	对象
metadata.name	集群的名称。集群的 DNS 记录是 **{{.metadata.name}}.{{.baseDomain}}** 的子域。	小写字母,连字符(-)和句点(.)的字符串，如 **dev**。该字符串长度必须为 14 个字符或更少。
platform	执行安装的具体平台配置：**aws**、**baremetal**、**azure**、**openstack**、**ovirt**、**vsphere**。有关 **platform.<platform>** 参数的额外信息，请参考下表来了解您的具体平台。	对象

参数	描述	值

pullSecret	从 Red Hat OpenShift Cluster Manager 获取 pull secret，验证从 Quay.io 等服务中下载 OpenShift Container Platform 组件的容器镜像。	<pre>{
		  "auths":{
		  "cloud.openshift.com":{
		  "auth":"b3Blb=",
		  "email":"you@example.com"
		  },
		  "quay.io":{
		  "auth":"b3Blb=",
		  "email":"you@example.com"
		  }
		  }
		}</pre>

9.2.10.2. 网络配置参数

您可以根据现有网络基础架构的要求自定义安装配置。例如，您可以扩展集群网络的 IP 地址块，或者提供不同于默认值的不同 IP 地址块。

只支持 IPv4 地址。

表 9.9. 网络参数

参数	描述	值
networking	集群网络的配置。	对象 **注意** 您不能在安装后修改 **networking** 对象指定的参数。
networking.networkType	要安装的集群网络供应商 Container Network Interface（CNI）插件。	**OpenShiftSDN** 或 **OVNKubernetes**。默认值为 **OpenShiftSDN**。

networking.clusterNetwork	pod 的 IP 地址块。
	
	默认值为 **10.128.0.0/14**，主机前缀为 **/23**。
	
	如果您指定多个 IP 地址块，则块不得互相重叠。
		一个对象数组。例如：
		
		<pre>networking:
		  clusterNetwork:
		  - cidr: 10.128.0.0/14
		  hostPrefix: 23</pre>

networking.clusterNetwork.cidr	使用 **networking.clusterNetwork** 时需要此项。IP 地址块。 一个 IPv4 网络。	使用 CIDR 形式的 IP 地址块。IPv4 块的前缀长度介于 **0** 到 **32** 之间。

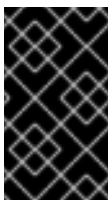
参数	描述	值
networking.clusterNetwork.hostPrefix	分配给每个单独节点的子网前缀长度。例如，如果 **hostPrefix** 设为 **23**，则每个节点从所给的 **cidr** 中分配一个 **/23** 子网。**hostPrefix** 值 **23** 提供 510（2^(32 - 23)- 2）个 pod IP 地址。	子网前缀。 默认值为 **23**。
networking.serviceNetwork	服务的 IP 地址块。默认值为 **172.30.0.0/16**。 OpenShift SDN 和 OVN-Kubernetes 网络供应商只支持服务网络的一个 IP 地址块。	CIDR 格式具有 IP 地址块的数组。例如： networking: serviceNetwork: - 172.30.0.0/16
networking.machineNetwork	机器的 IP 地址块。 如果您指定多个 IP 地址块，则块不得互相重叠。	一个对象数组。例如： networking: machineNetwork: - cidr: 10.0.0.0/16
networking.machineNetwork.cidr	使用 **networking.machineNetwork** 时需要。IP 地址块。libvirt 以外的所有平台的默认值为 **10.0.0.0/16**。对于 libvirt，默认值为 **192.168.126.0/24**。	CIDR 表示法中的 IP 网络块。 例如： **10.0.0.0/16**。 **注意** 将 **networking.machineNetwork** 设置为与首选 NIC 所在的 CIDR 匹配。

9.2.10.3. 可选配置参数

下表描述了可选安装配置参数：

表 9.10. 可选参数

参数	描述	值
additionalTrustBundle	添加到节点可信证书存储中的 PEM 编码 X.509 证书捆绑包。配置了代理时，也可以使用这个信任捆绑包。	字符串
compute	组成计算节点的机器的配置。	machine-pool 对象的数组。详情请查看以下"Machine-pool"表。

参数	描述	值
compute.architecture	决定池中机器的指令集合架构。目前不支持异构集群，因此所有池都必须指定相同的架构。有效值为 **amd64** （默认值）。	字符串
compute.hyperthreading	是否在计算机器上启用或禁用并发多线程或**超线程**。默认情况下，启用并发多线程以提高机器内核的性能。  **重要** 如果禁用并发多线程，请确保在容量规划时考虑到机器性能可能会显著降低的问题。	**Enabled** 或 **Disabled**
compute.name	使用 **compute** 时需要此值。机器池的名称。	**worker**
compute.platform	使用 **compute** 时需要此值。使用此参数指定托管 worker 机器的云供应商。此参数值必须与 **controlPlane.platform** 参数值匹配。	**aws**、**azure**、**gcp**、**openstack**、**ovirt**、**vsphere** 或 **{}**
compute.replicas	要置备的计算机器数量，也称为 worker 机器。	大于或等于 **2** 的正整数。默认值为 **3**。
controlPlane	组成 control plane 的机器的配置。	**MachinePool** 对象的数组。详情请查看以下"Machine-pool"表。
controlPlane.architecture	决定池中机器的指令集合架构。目前不支持异构集群，因此所有池都必须指定相同的架构。有效值为 **amd64** （默认值）。	字符串
controlPlane.hyperthreading	是否在 control plane 机器上启用或禁用并发多线程或**超线程**。默认情况下，启用并发多线程以提高机器内核的性能。  **重要** 如果禁用并发多线程，请确保在容量规划时考虑到机器性能可能会显著降低的问题。	**Enabled** 或 **Disabled**

参数	描述	值
controlPlane.name	使用 **controlPlane** 时需要。机器池的名称。	**master**
controlPlane.platform	使用 **controlPlane** 时需要。使用此参数指定托管 control plane 机器的云供应商。此参数值必须与 **compute.platform** 参数值匹配。	**aws**、**azure**、**gcp**、**openstack**、**ovirt**、**vsphere** 或 **{}**
controlPlane.replicas	要置备的 control plane 机器数量。	唯一支持的值是 **3**，它是默认值。
credentialsMode	Cloud Credential Operator（CCO）模式。如果没有指定任何模式，CCO 会动态地尝试决定提供的凭证的功能，在支持多个模式的平台上使用 mint 模式。 **注意** 不是所有 CCO 模式都支持所有云供应商。如需有关 CCO 模式的更多信息，请参阅*Red Hat Operator 参考指南*内容中的 *Cloud Credential Operator* 条目。	**Mint**、**Passthrough**、**Manual** 或空字符串(**""**)。
fips	启用或禁用 FIPS 模式。默认为 **false**（禁用）。如果启用了 FIPS 模式，运行 OpenShift Container Platform 的 Red Hat Enterprise Linux CoreOS (RHCOS) 机器会绕过默认的 Kubernetes 加密套件，并使用由 RHCOS 提供的加密模块。 **重要** 只有在 **x86_64** 架构中的 OpenShift Container Platform 部署支持 FIPS 验证的/Modules in Process 加密库。 **注意** 如果使用 Azure File 存储，则无法启用 FIPS 模式。	**false** 或 **true**

参数	描述	值
imageContentSources	release-image 内容的源和仓库。	对象数组。包括一个 **source** 以及可选的 **mirrors**，如下表所示。
imageContentSources.source	使用 **imageContentSources** 时需要。指定用户在镜像拉取规格中引用的仓库。	字符串
imageContentSources.mirrors	指定可能还包含同一镜像的一个或多个仓库。	字符串数组
publish	如何发布或公开集群的面向用户的端点，如 Kubernetes API、OpenShift 路由。	**Internal** 或 **External**。默认值为 **External**。 在非云平台上不支持将此字段设置为 **Internal**。 **重要** 如果将字段的值设为 **Internal**，集群将无法运行。如需更多信息，请参阅 BZ#1953035。
sshKey	用于验证集群机器访问的 SSH 密钥或密钥。 **注意** 对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 **ssh-agent** 进程使用的 SSH 密钥。	一个或多个密钥。例如： sshKey: <key1> <key2> <key3>

9.2.10.4. 其他 Red Hat OpenStack Platform（RHOSP）配置参数

下表描述了其他 RHOSP 配置参数：

表 9.11. 其他 RHOSP 参数

参数	描述	值
compute.platform.openstack.rootVolume.size	对于计算机器，以 GB 为单位表示的根卷大小。如果您不设置这个值，机器将使用临时存储。	整数，如 **30**。
compute.platform.openstack.rootVolume.type	对于计算机器，根卷的类型。	字符串，如 **performance**。
controlPlane.platform.openstack.rootVolume.size	对于 control plane 机器，以 GB 为单位表示的根卷大小。如果您不设置这个值，机器将使用临时存储。	整数，如 **30**。
controlPlane.platform.openstack.rootVolume.type	对于 control plane 机器，根卷的类型。	字符串，如 **performance**。
platform.openstack.cloud	要使用的 RHOSP 云的名称，来自于 **clouds.yaml** 文件中的云列表。	字符串，如 **MyCloud**。
platform.openstack.externalNetwork	用于安装的 RHOSP 外部网络名称。	字符串，如 **external**。
platform.openstack.computeFlavor	用于 control plane 和计算机器的 RHOSP 类别。	字符串，如 **m1.xlarge**。

9.2.10.5. 可选 RHOSP 配置参数

下表描述了可选 RHOSP 配置参数：

表 9.12. 可选的 RHOSP 参数

参数	描述	值
compute.platform.openstack.additionalNetworkIDs	与计算机器关联的其他网络。不能为额外网络创建允许的地址对。	一个或多个 UUID 列表作为字符串。例如：**fa806b2f-ac49-4bce-b9db-124bc64209bf**。
compute.platform.openstack.additionalSecurityGroupIDs	与计算机器关联的其他安全组。	一个或多个 UUID 列表作为字符串。例如：**7ee219f3-d2e9-48a1-96c2-e7429f1b0da7**。

参数	描述	值
compute.platform.openstack.zones	RHOSP Compute（Nova）可用区（AZ）在其中安装机器。如果没有设置此参数，安装程序会依赖于配置了 RHOSP 管理员的 Nova 的默认设置。 在使用 Kuryr 的集群上，RHOSP Octavia 不支持可用域。负载均衡器，如果您使用 Amphora 供应商驱动程序，则依赖 Amphora 虚拟机的 OpenShift Container Platform 服务不会根据此属性的值创建。	字符串列表。例如：**["zone-1", "zone-2"]**。
controlPlane.platform.openstack.additionalNetworkIDs	与 control plane 机器关联的额外网络。不能为额外网络创建允许的地址对。	一个或多个 UUID 列表作为字符串。例如：**fa806b2f-ac49-4bce-b9db-124bc64209bf**。
controlPlane.platform.openstack.additionalSecurityGroupIDs	与 control plane 机器关联的其他安全组。	一个或多个 UUID 列表作为字符串。例如：**7ee219f3-d2e9-48a1-96c2-e7429f1b0da7**。
controlPlane.platform.openstack.zones	RHOSP Compute（Nova）可用区（AZ）在其中安装机器。如果没有设置此参数，安装程序会依赖于配置了 RHOSP 管理员的 Nova 的默认设置。 在使用 Kuryr 的集群上，RHOSP Octavia 不支持可用域。负载均衡器，如果您使用 Amphora 供应商驱动程序，则依赖 Amphora 虚拟机的 OpenShift Container Platform 服务不会根据此属性的值创建。	字符串列表。例如：**["zone-1", "zone-2"]**。
platform.openstack.clusterOSImage	安装程序从中下载 RHCOS 镜像的位置。 您必须设置此参数以便在受限网络中执行安装。	HTTP 或 HTTPS URL，可选使用 SHA-256 checksum。 例如：**http://mirror.example.com/images/rhcos-43.81.201912131630.0-openstack.x86_64.qcow2.gz?sha256=ffebbd68e8a1f2a245ca19522c16c86f67f9ac8e4e0c1f0a812b068b16f7265d**。该值也可以是现有 Glance 镜像的名称，如 **my-rhcos**。

参数	描述	值

platform.openstack.defaultMachinePlatform	默认机器池平台配置。	<pre>{
		  "type": "ml.large",
		  "rootVolume": {
		  "size": 30,
		  "type": "performance"
		  }
		}</pre>

platform.openstack.ingressFloatingIP	与 Ingress 端口关联的现有浮动 IP 地址。要使用此属性，还必须定义 **platform.openstack.externalNetwork** 属性。	IP 地址，如 **128.0.0.1**。
platform.openstack.lbFloatingIP	与 API 负载均衡器关联的现有浮动 IP 地址。要使用此属性，还必须定义 **platform.openstack.externalNetwork** 属性。	IP 地址，如 **128.0.0.1**。
platform.openstack.externalDNS	集群实例用于进行 DNS 解析的外部 DNS 服务器的 IP 地址。	一个 IP 地址列表作为字符串。例如，**["8.8.8.8", "192.168.1.12"]**。
platform.openstack.machinesSubnet	集群节点使用的 RHOSP 子网的 UUID。在这个子网上创建节点和虚拟 IP（VIP）端口。 **networking.machineNetwork** 中的第一个项需要和 **machinesSubnet** 的值匹配。 如果部署到自定义子网中，则无法将外部 DNS 服务器指定到 OpenShift Container Platform 安装程序。反之，把 DNS 添加到 RHOSP 的子网。	作为字符串的 UUID。例如： **fa806b2f-ac49-4bce-b9db-124bc64209bf**。

9.2.10.6. RHOSP 部署中的自定义子网

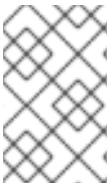
另外，您还可以在您选择的 Red Hat OpenStack Platform（RHOSP）子网中部署集群。子网的 GUID 作为 **install-config.yaml** 文件中的 **platform.openstack.machinesSubnet** 的值传递。

此子网被用作集群的主子网，在其上创建节点和端口。

在使用自定义子网运行 OpenShift Container Platform 安装程序前，请验证：

- 目标网络和子网可用。

- 目标子网上启用了 DHCP。
- 您可提供在目标网络上有创建端口权限的安装程序凭证。
- 如果您的网络配置需要一个路由器，它会在 RHOSP 中创建。有些配置依赖于路由器来转换浮动 IP 地址。
- 您的网络配置不依赖于供应商网络。不支持提供商网络。



注意

默认情况下，API VIP 使用 x.x.x.5，Ingress VIP 从网络 CIDR 块获取 x.x.x.7。要覆盖这些默认值，为 DHCP 分配池以外的 **platform.openstack.apiVIP** 和 **platform.openstack.ingressVIP** 设置值。

9.2.10.7. 使用 Kuryr 的 RHOSP 的自定义 install-config.yaml 文件示例

要使用 Kuryr SDN 而不是默认的 OpenShift SDN 部署，您必须修改 **install-config.yaml** 文件，使其包含 **Kuryr** 作为所需的 **networking.networkType**，然后执行默认的 OpenShift Container Platform SDN 安装步骤。此示例 **install-config.yaml** 展示了所有可能的 Red Hat OpenStack Platform (RHOSP) 自定义选项。

重要

此示例文件仅供参考。您必须使用安装程序来获取 **install-config.yaml** 文件。

```
apiVersion: v1
baseDomain: example.com
clusterID: os-test
controlPlane:
  name: master
  platform: {}
  replicas: 3
compute:
- name: worker
  platform:
    openstack:
      type: ml.large
  replicas: 3
metadata:
  name: example
networking:
  clusterNetwork:
  - cidr: 10.128.0.0/14
    hostPrefix: 23
  machineNetwork:
  - cidr: 10.0.0.0/16
  serviceNetwork:
  - 172.30.0.0/16 1
  networkType: Kuryr
platform:
  openstack:
    cloud: mycloud
    externalNetwork: external
```

```
    computeFlavor: m1.xlarge
    lbFloatingIP: 128.0.0.1
    trunkSupport: true 2
    octaviaSupport: true 3
pullSecret: '{"auths": ...}'
sshKey: ssh-ed25519 AAAA...
```

1 Amphora Octavia 驱动程序为每个负载均衡器创建两个端口。因此，安装程序创建的服务子网是由 **serviceNetwork** 属性值指定的 CIDR 的两倍。要防止 IP 地址冲突，则需要更大的范围。

2 3 安装程序会自动发现 **trunkSupport** 和 **octaviaSupport**，因此无需设置它们。但是，如果您的环境不满足这两个要求，Kuryr SDN 将无法正常工作。需要使用中继来把 pod 连接到 RHOSP 网络，并且需要 Octavia 来创建 OpenShift Container Platform 服务。

9.2.10.8. Kuryr 端口池

Kuryr 端口池在待机时维护多个端口，用于创建 pod。

将端口保留在待机上可最大程度缩短 pod 创建时间。如果没有端口池，Kuryr 必须明确请求在创建或删除 pod 时创建或删除端口。

Kuryr 使用的 Neutron 端口是在绑定到命名空间的子网中创建的。这些 pod 端口也作为子端口添加到 OpenShift Container Platform 集群节点的主端口。

因为 Kuryr 将每个命名空间保留在单独的子网中，所以对于每个“命名空间-worker”对都会维护一个单独的端口池。

在安装集群前，您可以在 **cluster-network-03-config.yml** 清单文件中设置以下参数来配置端口池行为：

- **enablePortPoolsPrepopulation** 参数控制池预填充，它会强制 Kuryr 在创建时（如添加新主机或创建新命名空间时）将端口添加到池中。默认值为 **false**。
- **poolMinPorts** 参数是池中保留的最少可用端口的数量。默认值为 **1**。
- **poolMaxPorts** 参数是池中保留的最大可用端口数。如果值为 **0**，会禁用上限。这是默认的设置。
  如果您的 OpenStack 端口配额较低，或者 pod 网络上的 IP 地址有限，请考虑设置此选项以确保删除不需要的端口。
- **poolBatchPorts** 参数定义一次可以创建的 Neutron 端口的最大数量。默认值为 **3**。

9.2.10.9. 在安装过程中调整 Kuryr 端口池

在安装过程中，您可以配置 Kuryr 如何管理 Red Hat OpenStack Platform (RHOSP) Neutron 端口，以控制 pod 创建的速度和效率。

先决条件

- 创建并修改 **install-config.yaml** 文件。

流程

1. 在命令行中创建清单文件：

```
$ ./openshift-install create manifests --dir <installation_directory> 1
```

1 对于 **<installation_directory>**，请指定含有集群的 **install-config.yaml** 文件的目录的名称。

2. 在 **<installation_directory>/manifests/** 目录下，创建一个名为 **cluster-network-03-config.yml** 的文件：

```
$ touch <installation_directory>/manifests/cluster-network-03-config.yml 1
```

1 对于 **<installation_directory>**，请指定包含集群的 **manifests/** 目录的目录名称。

创建该文件后，**manifests/** 目录中会包含多个网络配置文件，如下所示：

```
$ ls <installation_directory>/manifests/cluster-network-*
```

输出示例

```
cluster-network-01-crd.yml
cluster-network-02-config.yml
cluster-network-03-config.yml
```

3. 在编辑器中打开 **cluster-network-03-config.yml** 文件，并输入描述您想要的 Cluster Network Operator 配置的自定义资源(CR)：

```
$ oc edit networks.operator.openshift.io cluster
```

4. 编辑设置以满足您的要求。以下示例提供了以下文件：

```
apiVersion: operator.openshift.io/v1
kind: Network
metadata:
  name: cluster
spec:
  clusterNetwork:
  - cidr: 10.128.0.0/14
    hostPrefix: 23
  serviceNetwork:
  - 172.30.0.0/16
  defaultNetwork:
    type: Kuryr
    kuryrConfig:
      enablePortPoolsPrepopulation: false 1
      poolMinPorts: 1 2
      poolBatchPorts: 3 3
      poolMaxPorts: 5 4
      openstackServiceNetwork: 172.30.0.0/15 5
```

1 将 **enablePortPoolsPrepopulation** 的值设置为 **true** 以使 Kuryr 在创建命名空间或在集群中添加新节点后创建新 Neutron 端口。此设置引发 Neutron 端口配额，但可以缩短生成容器集所需的时间。默认值为 **false**。

2 如果池中的可用端口数量低于 **poolMinPorts** 的值，Kuryr 会为池创建新端口。默认值为 **1**。

3 **poolBatchPorts** 控制在可用端口数量低于 **poolMinPorts** 值时创建的新端口数量。默认值为 **3**。

4 如果池中的可用端口数量大于 **poolMaxPorts** 的值，Kuryr 会删除它们，直到数量与这个值匹配为止。将此值设置为 **0** 可禁用此上限，防止池缩小。默认值为 **0**。

5 **openStackServiceNetwork** 参数定义将 IP 地址分配到 RHOSP Octavia 的 LoadBalancer 的网络的 CIDR 范围。

如果此参数与 Amphora 驱动程序一起使用，则 Octavia 会为每个负载均衡器从这个网络获取两个 IP 地址：一个用于 OpenShift，另一个用于 VRRP 连接。由于这些 IP 地址分别由 OpenShift Container Platform 和 Neutron 管理，因此它们必须来自不同的池。因此，**openStackServiceNetwork** 的值必须至少是 **serviceNetwork** 值的两倍，**serviceNetwork** 的值必须与 **openStackServiceNetwork** 定义的范围完全重叠。

CNO 验证从此参数定义的范围获取的 VRRP IP 地址是否与 **serviceNetwork** 参数定义的范围不重叠。

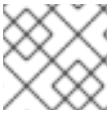
如果没有设置此参数，CNO 将使用 **serviceNetwork** 的扩展值，它是前缀大小值减 1。

5. 保存 **cluster-network-03-config.yml** 文件，再退出文本编辑器。
6. 可选：备份 **manifests/cluster-network-03-config.yml** 文件。安装程序在创建集群时删除 **manifests/** 目录。

9.2.11. 设置计算机器关联性

另外，您还可以在安装过程中为计算机器设置关联性策略。默认情况下，安装程序不会为计算机器选择关联性策略。

您还可以在安装后创建使用特定 RHOSP 服务器组的机器集。



注意

control plane 机器使用 **soft-anti-affinity** 策略创建。

提示

您可以在 RHOSP 文档中了解更多有关 RHOSP 实例调度和放置 的信息。

先决条件

- 创建 **install-config.yaml** 文件并完成对其所做的任何修改。

流程

1. 使用 RHOSP 命令行界面，为您的计算机器创建服务器组。例如：

```
$ openstack \
  --os-compute-api-version=2.15 \
  server group create \
```

```
    --policy anti-affinity \
    my-openshift-worker-group
```

如需更多信息，请参阅**服务器组 create** 命令文档。

2. 进入包含安装程序的目录并创建清单：

```
$ ./openshift-install create manifests --dir=<installation_directory>
```

其中：

installation_directory

指定包含集群的 **install-config.yaml** 文件的目录名称。

3. 打开 **manifests/99_openshift-cluster-api_worker-machineset-0.yaml**，这是 **MachineSet** 定义文件。

4. 将属性 **serverGroupID** 添加到 **spec.template.spec.providerSpec.value** 属性下的定义中。例如：

```
apiVersion: machine.openshift.io/v1beta1
kind: MachineSet
metadata:
  labels:
    machine.openshift.io/cluster-api-cluster: <infrastructure_ID>
    machine.openshift.io/cluster-api-machine-role: <node_role>
    machine.openshift.io/cluster-api-machine-type: <node_role>
  name: <infrastructure_ID>-<node_role>
  namespace: openshift-machine-api
spec:
  replicas: <number_of_replicas>
  selector:
    matchLabels:
      machine.openshift.io/cluster-api-cluster: <infrastructure_ID>
      machine.openshift.io/cluster-api-machineset: <infrastructure_ID>-<node_role>
  template:
    metadata:
      labels:
        machine.openshift.io/cluster-api-cluster: <infrastructure_ID>
        machine.openshift.io/cluster-api-machine-role: <node_role>
        machine.openshift.io/cluster-api-machine-type: <node_role>
        machine.openshift.io/cluster-api-machineset: <infrastructure_ID>-<node_role>
    spec:
      providerSpec:
        value:
          apiVersion: openstackproviderconfig.openshift.io/v1alpha1
          cloudName: openstack
          cloudsSecret:
            name: openstack-cloud-credentials
            namespace: openshift-machine-api
          flavor: <nova_flavor>
          image: <glance_image_name_or_location>
          serverGroupID: aaaaaaaa-bbbb-cccc-dddd-eeeeeeeeeeee (1)
          kind: OpenstackProviderSpec
          networks:
```

```
    - filter: {}
      subnets:
      - filter:
          name: <subnet_name>
          tags: openshiftClusterID=<infrastructure_ID>
    securityGroups:
    - filter: {}
      name: <infrastructure_ID>-<node_role>
    serverMetadata:
      Name: <infrastructure_ID>-<node_role>
      openshiftClusterID: <infrastructure_ID>
    tags:
    - openshiftClusterID=<infrastructure_ID>
    trunk: true
    userDataSecret:
      name: <node_role>-user-data
    availabilityZone: <optional_openstack_availability_zone>
```

1 在此处添加服务器组的 UUID。

5. 可选：备份 **manifests/99_openshift-cluster-api_worker-machineset-0.yaml** 文件。创建集群时，安装程序会删除 **manifests/** 目录。

安装集群时，安装程序将使用您修改的 **MachineSet** 定义在 RHOSP 服务器组中创建计算机器。

9.2.12. 生成 SSH 私钥并将其添加到代理中

如果要在集群上执行安装调试或灾难恢复，则必须为 **ssh-agent** 和安装程序提供 SSH 密钥。您可以使用此密钥访问公共集群中的 bootstrap 机器来排除安装问题。

注意

在生产环境中，您需要进行灾难恢复和调试。

您可以使用此密钥以 **core** 用户身份通过 SSH 连接到 master 节点。在部署集群时，此密钥会添加到 **core** 用户的 **~/.ssh/authorized_keys** 列表中。

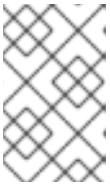
流程

1. 如果还没有为计算机上免密码身份验证而配置的 SSH 密钥，请创建一个。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ ssh-keygen -t ed25519 -N '' \
  -f <path>/<file_name> 1
```

1 指定新 SSH 密钥的路径和文件名，如 **~/.ssh/id_rsa**。如果您已有密钥对，请确保您的公钥位于 **~/.ssh** 目录中。

运行此命令会在指定的位置生成不需要密码的 SSH 密钥。



注意

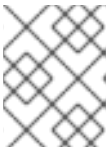
如果您计划在 **x86_64** 架构中安装使用 FIPS 验证的/Modules in Process 加密库的 OpenShift Container Platform 集群，不要创建使用 **ed25519** 算法的密钥。反之，创建一个使用 **rsa** 或 **ecdsa** 算法的密钥。

2. 作为后台任务启动 **ssh-agent** 进程：

```
$ eval "$(ssh-agent -s)"
```

输出示例

```
Agent pid 31874
```



注意

如果您的集群采用 FIPS 模式，则只使用 FIPS 兼容算法来生成 SSH 密钥。密钥必须是 RSA 或 ECDSA。

3. 将 SSH 私钥添加到 **ssh-agent**：

```
$ ssh-add <path>/<file_name> (1)
```

输出示例

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

(1) 指定 SSH 私钥的路径和文件名，如 **~/.ssh/id_rsa**

后续步骤

- 在安装 OpenShift Container Platform 时，为安装程序提供 SSH 公钥。

9.2.13. 启用对环境的访问

在部署时，所有 OpenShift Container Platform 机器都是在 Red Hat OpenStack Platform (RHOSP) 租户网络中创建的。因此，大多数 RHOSP 部署中都无法直接访问它们。

您可以在安装过程中使用浮动 IP 地址（FIP）来配置 OpenShift Container Platform API 和应用程序访问。您也可以在没有配置 FIP 的情况下完成安装，但安装程序不会配置一种从外部访问 API 或应用程序的方法。

9.2.13.1. 启用通过浮动 IP 地址进行访问

创建浮动 IP（FIP）地址，用于从外部访问 OpenShift Container Platform API 和集群应用程序。

流程

1. 使用 Red Hat OpenStack Platform (RHOSP) CLI，创建 API FIP：

```
$ openstack floating ip create --description "API <cluster_name>.<base_domain>"
<external_network>
```

2. 使用 Red Hat OpenStack Platform (RHOSP) CLI，创建应用程序或 Ingress，FIP：

```
$ openstack floating ip create --description "Ingress <cluster_name>.<base_domain>"
<external_network>
```

3. 向用于 API 和 Ingress FIP 的 DNS 服务器添加符合这些模式的记录：

```
api.<cluster_name>.<base_domain>.  IN  A  <API_FIP>
*.apps.<cluster_name>.<base_domain>. IN  A <apps_FIP>
```

注意

如果您不控制 DNS 服务器，您可以通过将集群域名（如以下内容）添加到 **/etc/hosts** 文件中来访问集群：

- **<api_floating_ip> api.<cluster_name>.<base_domain>**
- **<application_floating_ip> grafana-openshift-monitoring.apps.<cluster_name>.<base_domain>**
- **<application_floating_ip> prometheus-k8s-openshift-monitoring.apps.<cluster_name>.<base_domain>**
- **<application_floating_ip> oauth-openshift.apps.<cluster_name>.<base_domain>**
- **<application_floating_ip> console-openshift-console.apps.<cluster_name>.<base_domain>**
- **application_floating_ip integrate-oauth-server-openshift-authentication.apps.<cluster_name>.<base_domain>**

/etc/hosts 文件中的集群域名授予对本地集群的 Web 控制台和监控界面的访问权限。您还可以使用 **kubectl** 或 **oc**。您可以使用指向 <application_floating_ip> 的额外条目来访问用户应用程序。此操作使 API 和应用程序可供您访问，不适合于生产部署，但允许对开发和测试进行安装。

4. 将 FIP 添加到 **install-config.yaml** 文件，将其作为以下参数的值：
   - **platform.openstack.ingressFloatingIP**
   - **platform.openstack.lbFloatingIP**

如果使用这些值，还必须在 **install-config.yaml** 文件中输入一个外部网络作为 **platform.openstack.externalNetwork** 参数的值。

提示

您可以通过分配浮动 IP 地址并更新防火墙配置，使 OpenShift Container Platform 资源在集群之外可用。

9.2.13.2. 完成没有浮动 IP 地址的安装

您可以在不提供浮动 IP 地址的情况下在 Red Hat OpenStack Platform（RHOSP）上安装 OpenShift Container Platform。

在 **install-config.yaml** 文件中，不要定义以下参数：

- **platform.openstack.ingressFloatingIP**
- **platform.openstack.lbFloatingIP**

如果您无法提供外部网络，也可以将 **platform.openstack.externalNetwork** 留空。如果没有为 **platform.openstack.externalNetwork** 提供值，则不会为您创建路由器。如果没有额外的操作，安装程序将无法从 Glance 检索镜像。您必须自行配置外部连接。

如果在因为缺少浮动 IP 地址或名称解析而无法访问集群 API 的系统中运行安装程序时，安装会失败。要防止安装失败，可以使用代理网络或者从与您的机器位于同一网络的系统中运行安装程序。

注意

您可以通过为 API 和 Ingress 端口创建 DNS 记录来启用名称解析。例如：

```
api.<cluster_name>.<base_domain>.  IN  A  <api_port_IP>
*.apps.<cluster_name>.<base_domain>. IN  A <ingress_port_IP>
```

如果您不控制 DNS 服务器，可以改为将记录添加到 **/etc/hosts** 文件中。此操作使 API 可供您自己访问，不适合于生产部署。这可用于进行开发和测试的安装。

9.2.14. 部署集群

您可以在兼容云平台中安装 OpenShift Container Platform。

重要

安装程序的 **create cluster** 命令只能在初始安装过程中运行一次。

先决条件

- 获取 OpenShift Container Platform 安装程序以及集群的 pull secret。

流程

1. 更改为包含安装程序的目录并初始化集群部署：

```
$ ./openshift-install create cluster --dir <installation_directory> \ 1
    --log-level=info 2
```

1. 对于 **<installation_directory>**，请指定自定义 **./install-config.yaml** 文件的位置。
2. 要查看不同的安装详情，请指定 **warn**、**debug** 或 **error**，而不要指定 **info**。

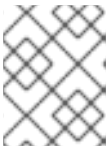
注意

如果您在主机上配置的云供应商帐户没有足够的权限来部署集群，安装过程将会停止，并且显示缺少的权限。

集群部署完成后，终端会显示访问集群的信息，包括指向其 Web 控制台的链接和 **kubeadmin** 用户的凭证。

输出示例

```
...
INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run 'export
KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'
INFO Access the OpenShift web-console here: https://console-openshift-
console.apps.mycluster.example.com
INFO Login to the console with user: "kubeadmin", and password: "4vYBz-Ee6gm-ymBZj-
Wt5AL"
INFO Time elapsed: 36m22s
```



注意

当安装成功时，集群访问和凭证信息还会输出到 **<installation_directory>/.openshift_install.log**。

重要

- 安装程序生成的 Ignition 配置文件包含在 24 小时后过期的证书，然后在过期时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外情况是，您需要手动批准待处理的 **node-bootstrapper** 证书签名请求（CSR）来恢复 kubelet 证书。如需更多信息，请参阅*从过期的 control plane 证书中恢复*的文档。
- 建议您在生成 12 小时后使用 Ignition 配置文件，因为集群安装后 24 小时证书从 16 小时轮转至 22 小时。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中运行证书更新时避免安装失败。

重要

您不得删除安装程序或安装程序所创建的文件。需要这两者才能删除集群。

9.2.15. 验证集群状态

您可以在安装过程中或安装后验证 OpenShift Container Platform 集群的状态：

流程

1. 在集群环境中，导出管理员的 kubeconfig 文件：

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

kubeconfig 文件包含关于集群的信息，供 CLI 用于将客户端连接到正确集群和 API 服务器。

2. 查看部署后创建的 control plane 和计算机器：

```
$ oc get nodes
```

3. 查看集群的版本：

```
$ oc get clusterversion
```

4. 查看 Operator 的状态：

```
$ oc get clusteroperator
```

5. 查看集群中的所有正在运行的 pod:

```
$ oc get pods -A
```

9.2.16. 使用 CLI 登录到集群

您可以通过导出集群 **kubeconfig** 文件，以默认系统用户身份登录集群。**kubeconfig** 文件包含关于集群的信息，供 CLI 用于将客户端连接到正确集群和 API 服务器。该文件特只适用于一个特定的集群，在 OpenShift Container Platform 安装过程中创建。

先决条件

- 已部署了 OpenShift Container Platform 集群。
- 已安装 **oc** CLI。

流程

1. 导出 **kubeadmin** 凭证：

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

2. 使用导出的配置，验证能否成功运行 **oc** 命令：

```
$ oc whoami
```

输出示例

```
system:admin
```

其他资源

- 如需有关访问和了解 OpenShift Container Platform Web 控制台的更多信息，请参阅访问 Web 控制台。

9.2.17. OpenShift Container Platform 的 Telemetry 访问

在 OpenShift Container Platform 4.6 中，默认运行的 Telemetry 服务提供有关集群健康状况和成功更新的指标，需要访问互联网。如果您的集群连接到互联网，Telemetry 会自动运行，而且集群会注册到 OpenShift Cluster Manager。

确认 OpenShift Cluster Manager 清单正确后，可以由 Telemetry 自动维护，也可以使用 OpenShift Cluster Manager 手动维护，使用订阅监控来跟踪帐户或多集群级别的 OpenShift Container Platform 订阅。

其他资源

- 有关 Telemetry 服务的更多信息，请参阅关于远程健康监控。

9.2.18. 后续步骤

- 自定义集群。
- 如果需要，您可以选择不使用远程健康报告。
- 如果您需要启用对节点端口的外部访问，请使用节点端口配置集群流量。
- 如果您没有将 RHOSP 配置为使用浮动 IP 地址接受应用程序流量，使用浮动 IP 地址配置 RHOSP 访问。

9.3. 在您自己的基础架构的 OPENSTACK 上安装集群

在 OpenShift Container Platform 版本 4.6 中，您可以在运行于用户置备的基础架构上的 Red Hat OpenStack Platform（RHOSP）上安装集群。

通过利用您自己的基础架构，您可以将集群与现有的基础架构进行集成。和安装程序置备的安装方式相比，这个过程需要用户进行更多操作，因为您必须创建所有 RHOSP 资源，如 Nova 服务器、Neutron 端口和安全组。红帽提供了 Ansible playbook 来帮助您完成部署过程。

9.3.1. 先决条件

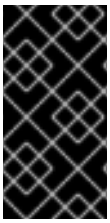
- 查看有关 OpenShift Container Platform 安装和更新流程的详细信息。
    - 在 *Available platforms* 部分验证 OpenShift Container Platform 4.6 是否与您的 RHOSP 版本兼容。您还可以查看 OpenShift Container Platform 在 RHOSP 中的支持 来比较不同版本的平台支持。
- 验证您的网络配置不依赖于供应商网络。不支持提供商网络。
- 具有要安装 OpenShift Container Platform 的 RHOSP 帐户
- 在您运行安装程序的机器中，有：
    - 用来保存在安装过程中创建的文件的一个单一目录
    - Python 3

9.3.2. OpenShift Container Platform 的互联网访问

在 OpenShift Container Platform 4.6 中，您需要访问互联网来安装集群。

您必须具有以下互联网访问权限：

- 访问 OpenShift Cluster Manager 以下载安装程序并执行订阅管理。如果集群可以访问互联网，并且没有禁用 Telemetry，该服务会自动授权您的集群。
- 访问 Quay.io，以获取安装集群所需的软件包。
- 获取执行集群更新所需的软件包。



重要

如果您的集群无法直接访问互联网，则可以在置备的某些类基础架构上执行受限网络安装。在此过程中，您要下载所需的内容，并使用它在镜像 registry（mirror registry） 中填充安装集群并生成安装程序所需的软件包。对于某些安装类型，集群要安装到的环境不需要访问互联网。在更新集群之前，要更新 registry 镜像系统中的内容。

9.3.3. 在 RHOSP 上安装 OpenShift Container Platform 的资源指南

您的 Red Hat OpenStack Platform（RHOSP）配额需要满足以下条件才支持 OpenShift Container Platform 安装：

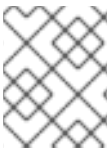
表 9.13. RHOSP 上默认 OpenShift Container Platform 集群的建议资源

资源	值
浮动 IP 地址	3
端口	15
路由器	1
子网	1
RAM	112 GB
vCPUs	28
卷存储	275 GB
实例	7
安全组	3
安全组规则	60

集群或许能使用少于推荐数量的资源来运作，但其性能无法保证。

重要

如果 RHOSP 对象存储（Swift）可用，并由具有 **swiftoperator** 角色的用户帐户执行，它会作为 OpenShift Container Platform 镜像 registry 的默认后端。在这种情况下，卷存储需要有 175GB。根据镜像 registry 的大小，Swift 空间要求会有所不同。



注意

默认情况下，您的安全组和安全组规则配额可能较低。如果遇到问题，请以 admin 的身份运行 **openstack quota set --secgroups 3 --secgroup-rules 60 <project>** 来提高配额。

OpenShift Container Platform 部署由 control plane 机器、计算机器和 bootstrap 机器组成。

9.3.3.1. control plane 机器

默认情况下，OpenShift Container Platform 安装过程会创建三台 control plane 机器。

每台机器都需要：

- 来自 RHOSP 配额的实例
- 来自 RHOSP 配额的端口
- 至少 16 GB 内存、4 个 vCPU 和 100 GB 存储空间的类别

9.3.3.2. 计算机器

默认情况下，OpenShift Container Platform 安装过程会创建三台计算机器。

每台机器都需要：

- 来自 RHOSP 配额的实例
- 来自 RHOSP 配额的端口
- 至少有 8 GB 内存、2 个 vCPU 和 100 GB 存储空间的类别

提示

计算机器托管您在 OpenShift Container Platform 上运行的应用程序；运行数量应尽可能多。

9.3.3.3. bootstrap 机器

在安装时，会临时置备 bootstrap 机器来支持 control plane。生产控制平面就绪后，bootstrap 机器会被取消置备。

bootstrap 机器需要：

- 来自 RHOSP 配额的实例
- 来自 RHOSP 配额的端口
- 至少 16 GB 内存、4 个 vCPU 和 100 GB 存储空间的类别

9.3.4. 下载 playbook 的依赖项

简化用户置备基础架构安装过程的 Ansible playbook 需要几个 Python 模块。在您要运行安装程序的机器上添加模块的仓库，然后下载它们。

注意

这些说明假设您使用 Red Hat Enterprise Linux（RHEL）8。

先决条件

- Python 3 已安装在您的机器上。

流程

1. 在命令行中添加软件仓库：
   a. 使用 Red Hat Subscription Manager 注册：

```
$ sudo subscription-manager register # If not done already
```

   b. 获取最新的订阅数据：

```
$ sudo subscription-manager attach --pool=$YOUR_POOLID # If not done already
```

   c. 禁用当前的软件仓库：

```
$ sudo subscription-manager repos --disable=* # If not done already
```

   d. 添加所需的软件仓库：

```
$ sudo subscription-manager repos \
  --enable=rhel-8-for-x86_64-baseos-rpms \
  --enable=openstack-16-tools-for-rhel-8-x86_64-rpms \
  --enable=ansible-2.9-for-rhel-8-x86_64-rpms \
  --enable=rhel-8-for-x86_64-appstream-rpms
```

2. 安装模块：

```
$ sudo yum install python3-openstackclient ansible python3-openstacksdk python3-netaddr
```

3. 确保 **python** 命令指向 **python3**:

```
$ sudo alternatives --set python /usr/bin/python3
```

9.3.5. 下载安装 playbook

下载 Ansible playbook，可用于在您自己的 Red Hat OpenStack Platform（RHOSP）基础架构上安装 OpenShift Container Platform。

先决条件

- curl 命令行工具可在您的机器上找到。

流程

流程

- 要将 playbook 下载到您的工作目录中，请从命令行运行以下脚本：

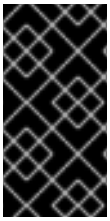
```
$ xargs -n 1 curl -O <<< '
    https://raw.githubusercontent.com/openshift/installer/release-4.6/upi/openstack/bootstrap.yaml
    https://raw.githubusercontent.com/openshift/installer/release-4.6/upi/openstack/common.yaml
    https://raw.githubusercontent.com/openshift/installer/release-4.6/upi/openstack/compute-nodes.yaml
    https://raw.githubusercontent.com/openshift/installer/release-4.6/upi/openstack/control-plane.yaml
    https://raw.githubusercontent.com/openshift/installer/release-4.6/upi/openstack/inventory.yaml
    https://raw.githubusercontent.com/openshift/installer/release-4.6/upi/openstack/network.yaml
    https://raw.githubusercontent.com/openshift/installer/release-4.6/upi/openstack/security-groups.yaml
    https://raw.githubusercontent.com/openshift/installer/release-4.6/upi/openstack/down-bootstrap.yaml
    https://raw.githubusercontent.com/openshift/installer/release-4.6/upi/openstack/down-compute-nodes.yaml
    https://raw.githubusercontent.com/openshift/installer/release-4.6/upi/openstack/down-control-plane.yaml
    https://raw.githubusercontent.com/openshift/installer/release-4.6/upi/openstack/down-load-balancers.yaml
    https://raw.githubusercontent.com/openshift/installer/release-4.6/upi/openstack/down-network.yaml
    https://raw.githubusercontent.com/openshift/installer/release-4.6/upi/openstack/down-security-groups.yaml
    https://raw.githubusercontent.com/openshift/installer/release-4.6/upi/openstack/down-containers.yaml'
```

playbook 下载到您的机器中。

重要

在安装过程中，您可以修改 playbook 来配置部署。

在集群生命周期中保留所有 playbook。您必须具有 playbook，才能从 RHOSP 中删除 OpenShift Container Platform 集群。



重要

您在 **bootstrap.yaml**、**compute-nodes.yaml**、**control-plane.yaml**、**network.yaml** 和 **security-groups.yaml** 文件中进行的任何改变都需要与带有 **down-** 前缀的对应的 playbook 相匹配。例如，对 **bootstrap.yaml** 文件的编辑也必须反映在 **down-bootstrap.yaml** 文件中。如果没有编辑这两个文件，则支持的删除集群过程将失败。

9.3.6. 获取安装程序

在安装 OpenShift Container Platform 之前，将安装文件下载到本地计算机上。

先决条件

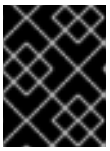
- 运行 Linux 或 macOS 的计算机，本地磁盘空间为 500 MB

流程

1. 访问 OpenShift Cluster Manager 站点的 Infrastructure Provider 页面。如果您有红帽帐号，请使用自己的凭证登录。如果没有，请创建一个帐户。
2. 选择您的基础架构供应商。
3. 进入适用于您的安装类型的页面，下载您的操作系统的安装程序，并将文件放在要保存安装配置文件的目录中。。

重要

安装程序会在用来安装集群的计算机上创建若干文件。在完成集群安装后，您必须保留安装程序和安装程序所创建的文件。这两个文件都需要删除集群。



重要

删除安装程序创建的文件不会删除您的集群，即使集群在安装过程中失败也是如此。要删除集群，为特定云供应商完成 OpenShift Container Platform 卸载流程。

4. 提取安装程序。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ tar xvf openshift-install-linux.tar.gz
```

5. 从 Red Hat OpenShift Cluster Manager 下载安装 pull secret 。通过此 pull secret，您可以进行所含授权机构提供的服务的身份验证，这些服务包括为 OpenShift Container Platform 组件提供容器镜像的 Quay.io。

9.3.7. 生成 SSH 私钥并将其添加到代理中

如果要在集群上执行安装调试或灾难恢复，则必须为 **ssh-agent** 和安装程序提供 SSH 密钥。您可以使用此密钥访问公共集群中的 bootstrap 机器来排除安装问题。

注意

在生产环境中，您需要进行灾难恢复和调试。

您可以使用此密钥以 **core** 用户身份通过 SSH 连接到 master 节点。在部署集群时，此密钥会添加到 **core** 用户的 **~/.ssh/authorized_keys** 列表中。

注意

您必须使用一个本地密钥，而不要使用在特定平台上配置的密钥，如 AWS 密钥对。

流程

1. 如果还没有为计算机上免密码身份验证而配置的 SSH 密钥，请创建一个。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ ssh-keygen -t ed25519 -N '' \
    -f <path>/<file_name> 1
```

1 指定新 SSH 密钥的路径和文件名，如 ~/**.ssh/id_rsa**。如果您已有密钥对，请确保您的公钥位于 ~/**.ssh** 目录中。

运行此命令会在指定的位置生成不需要密码的 SSH 密钥。

注意

如果您计划在 **x86_64** 架构中安装使用 FIPS 验证的/Modules in Process 加密库的 OpenShift Container Platform 集群，不要创建使用 **ed25519** 算法的密钥。反之，创建一个使用 **rsa** 或 **ecdsa** 算法的密钥。

2. 作为后台任务启动 **ssh-agent** 进程：

```
$ eval "$(ssh-agent -s)"
```

输出示例

```
Agent pid 31874
```

注意

如果您的集群采用 FIPS 模式，则只使用 FIPS 兼容算法来生成 SSH 密钥。密钥必须是 RSA 或 ECDSA。

3. 将 SSH 私钥添加到 **ssh-agent**：

```
$ ssh-add <path>/<file_name> 1
```

输出示例

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

1 指定 SSH 私钥的路径和文件名，如 ~/**.ssh/id_rsa**

后续步骤

- 在安装 OpenShift Container Platform 时，为安装程序提供 SSH 公钥。

9.3.8. 创建 Red Hat Enterprise Linux CoreOS (RHCOS) 镜像

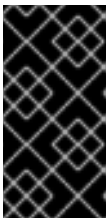
OpenShift Container Platform 安装程序要求 Red Hat OpenStack Platform (RHOSP) 集群中有 Red Hat Enterprise Linux CoreOS (RHCOS) 镜像。检索最新的 RHCOS 镜像，然后使用 RHOSP CLI 上传该镜像。

先决条件

- 已安装了 RHOSP CLI。

流程

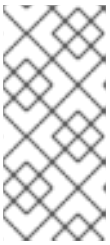
1. 登录到红帽客户门户网站的产品下载页。
2. 在 **Version** 下，为 Red Hat Enterprise Linux（RHEL）8 选择 OpenShift Container Platform 4.6 的最新发行版本。



重要

RHCOS 镜像可能不会随着 OpenShift Container Platform 的每一发行版本都有改变。您必须下载最高版本的镜像，其版本号应小于或等于您安装的 OpenShift Container Platform 版本。如果可用，请使用与 OpenShift Container Platform 版本匹配的镜像版本。

3. 下载 *Red Hat Enterprise Linux CoreOS（RHCOS） - OpenStack Image (QCOW)* 。
4. 解压镜像。



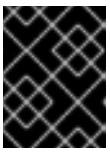
注意

您必须解压 RHOSP 镜像，然后集群才能使用它。下载的文件名可能不包含压缩扩展名，如 **.gz** 或 **.tgz**。要找出是否或者如何压缩文件，请在命令行中输入：

```
$ file <name_of_downloaded_file>
```

5. 从您下载的镜像，使用 RHOSP CLI 在集群中创建名为 **rhcos** 的镜像：

```
$ openstack image create --container-format=bare --disk-format=qcow2 --file rhcos-${RHCOS_VERSION}-openstack.qcow2 rhcos
```



重要

根据您的 RHOSP 环境，可能需要使用 **.raw** 或 **.qcow2** 格式下载镜像。如果使用 Ceph，则必须使用 **.raw** 格式。

警告

如果安装程序发现多个同名的镜像，它会随机选择其中之一。为避免这种行为，请在 RHOSP 中为资源创建唯一名称。

将镜像上传到 RHOSP 后，就可以被安装程序使用。

9.3.9. 验证外部网络访问

OpenShift Container Platform 安装进程需要外部网络访问权限。您必须为其提供外部网络值，否则部署会失败。在运行安装进程前，请验证 Red Hat OpenStack Platform (RHOSP) 中是否存在具有外部路由器类型的网络。

先决条件

- 将 OpenStack 联网服务配置为使用 DHCP 代理转发实例 DNS 查询

流程

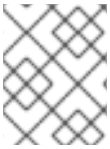
1. 使用 RHOSP CLI 验证"外部"网络的名称和 ID：

```
$ openstack network list --long -c ID -c Name -c "Router Type"
```

输出示例

```
+--------------------------------------+----------------+-------------+
| ID                                   | Name           | Router Type |
+--------------------------------------+----------------+-------------+
| 148a8023-62a7-4672-b018-003462f8d7dc | public_network | External    |
+--------------------------------------+----------------+-------------+
```

网络列表中会显示具有外部路由器类型的网络。如果最少有一个没有，请参阅 创建默认浮动 IP 网络和创建默认供应商网络。



注意

如果启用了 Neutron 中继服务插件，则默认创建中继端口。如需更多信息，请参阅 Neutron 中继端口。

9.3.10. 启用对环境的访问

在部署时，所有 OpenShift Container Platform 机器都是在 Red Hat OpenStack Platform (RHOSP) 租户网络中创建的。因此，大多数 RHOSP 部署中都无法直接访问它们。

您可以在安装过程中使用浮动 IP 地址（FIP）来配置 OpenShift Container Platform API 和应用程序访问。您也可以在没有配置 FIP 的情况下完成安装，但安装程序不会配置一种从外部访问 API 或应用程序的方法。

9.3.10.1. 启用通过浮动 IP 地址进行访问

创建浮动 IP(FIP)地址，用于从外部访问 OpenShift Container Platform API、集群应用程序和 bootstrap 过程。

流程

1. 使用 Red Hat OpenStack Platform (RHOSP) CLI，创建 API FIP：

```
$ openstack floating ip create --description "API <cluster_name>.<base_domain>"
<external_network>
```

2. 使用 Red Hat OpenStack Platform (RHOSP) CLI，创建应用程序或 Ingress，FIP：

```
$ openstack floating ip create --description "Ingress <cluster_name>.<base_domain>"
<external_network>
```

3. 使用 Red Hat OpenStack Platform（RHOSP）CLI 创建 bootstrap FIP:

```
$ openstack floating ip create --description "bootstrap machine" <external_network>
```

4. 向用于 API 和 Ingress FIP 的 DNS 服务器添加符合这些模式的记录：

```
api.<cluster_name>.<base_domain>.  IN  A  <API_FIP>
*.apps.<cluster_name>.<base_domain>. IN  A <apps_FIP>
```

注意

如果您不控制 DNS 服务器，您可以通过将集群域名（如以下内容）添加到 **/etc/hosts** 文件中来访问集群：

- **<api_floating_ip> api.<cluster_name>.<base_domain>**
- **<application_floating_ip> grafana-openshift-monitoring.apps.<cluster_name>.<base_domain>**
- **<application_floating_ip> prometheus-k8s-openshift-monitoring.apps.<cluster_name>.<base_domain>**
- **<application_floating_ip> oauth-openshift.apps.<cluster_name>.<base_domain>**
- **<application_floating_ip> console-openshift-console.apps.<cluster_name>.<base_domain>**
- **application_floating_ip integrate-oauth-server-openshift-authentication.apps.<cluster_name>.<base_domain>**

/etc/hosts 文件中的集群域名授予对本地集群的 Web 控制台和监控界面的访问权限。您还可以使用 **kubectl** 或 **oc**。您可以使用指向 <application_floating_ip> 的额外条目来访问用户应用程序。此操作使 API 和应用程序可供您访问，不适合于生产部署，但允许对开发和测试进行安装。

5. 将 FIP 添加到 **inventory.yaml** 文件，作为以下变量的值：
   - **os_api_fip**
   - **os_bootstrap_fip**
   - **os_ingress_fip**

如果使用这些值，还必须在 **inventory.yaml** 文件中输入一个外部网络作为 **os_external_network** 变量的值。

提示

您可以通过分配浮动 IP 地址并更新防火墙配置，使 OpenShift Container Platform 资源在集群之外可用。

9.3.10.2. 完成没有浮动 IP 地址的安装

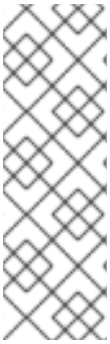
您可以在不提供浮动 IP 地址的情况下在 Red Hat OpenStack Platform（RHOSP）上安装 OpenShift Container Platform。

在 **inventory.yaml** 文件中，不要定义以下变量：

- **os_api_fip**
- **os_bootstrap_fip**
- **os_ingress_fip**

如果无法提供外部网络，也可以将 **os_external_network** 留空。如果没有为 **os_external_network** 提供值，则不会为您创建路由器。如果没有额外的操作，安装程序将无法从 Glance 检索镜像。之后在安装过程中，当您创建网络资源时，必须自行配置外部连接。

如果您使用 **wait-for** 命令从因为缺少浮动 IP 地址或名称解析而无法访问集群 API 的系统中运行安装程序时，安装会失败。要防止安装失败，可以使用代理网络或者从与您的机器位于同一网络的系统中运行安装程序。



注意

您可以通过为 API 和 Ingress 端口创建 DNS 记录来启用名称解析。例如：

```
api.<cluster_name>.<base_domain>.  IN  A  <api_port_IP>
*.apps.<cluster_name>.<base_domain>. IN  A <ingress_port_IP>
```

如果您不控制 DNS 服务器，可以改为将记录添加到 **/etc/hosts** 文件中。此操作使 API 可供您自己访问，不适合于生产部署。这可用于进行开发和测试的安装。

9.3.11. 为安装程序定义参数

OpenShift Container Platform 安装程序依赖于一个名为 **clouds.yaml** 的文件。该文件描述了 Red Hat OpenStack Platform (RHOSP) 配置参数，包括项目名称、登录信息和授权服务 URL。

流程

1. 创建 **clouds.yaml** 文件：
   - 如果您的 RHOSP 发行版包含 Horizon web UI，请在该 UI 中生成 **clouds.yaml** 文件。

     重要

     请记住在 **auth** 字段中添加密码。您也可以把 secret 保存在 **clouds.yaml** 以外的一个独立的文件中。

   - 如果您的 RHOSP 发行版不包含 Horizon Web UI，或者您不想使用 Horizon，请自行创建该文件。如需有关 **clouds.yaml** 的详细信息，请参阅 RHOSP 文档中的配置文件。

     ```
     clouds:
       shiftstack:
         auth:
           auth_url: http://10.10.14.42:5000/v3
           project_name: shiftstack
           username: shiftstack_user
           password: XXX
           user_domain_name: Default
           project_domain_name: Default
     ```

```
dev-env:
  region_name: RegionOne
  auth:
    username: 'devuser'
    password: XXX
    project_name: 'devonly'
    auth_url: 'https://10.10.14.22:5001/v2.0'
```

2. 如果您的 RHOSP 安装使用自签名证书颁发机构 (CA) 证书进行端点身份验证：

   a. 将 CA 文件复制到您的机器中。

   b. 将机器添加到证书颁发机构信任捆绑包中：

   ```
   $ sudo cp ca.crt.pem /etc/pki/ca-trust/source/anchors/
   ```

   c. 更新信任捆绑包：

   ```
   $ sudo update-ca-trust extract
   ```

   d. 将 **cacerts** 键添加到 **clouds.yaml** 文件。该值必须是到 CA 证书的绝对路径，则其可以被非根用户访问：

   ```
   clouds:
     shiftstack:
       ...
       cacert: "/etc/pki/ca-trust/source/anchors/ca.crt.pem"
   ```

   提示

   使用自定义 CA 证书运行安装程序后，您可以通过编辑 **cloud-provider-config** keymap 中的**ca-cert.pem** 键的值来更新证书。在命令行中运行：

   ```
   $ oc edit configmap -n openshift-config cloud-provider-config
   ```

3. 将 **clouds.yaml** 文件放在以下位置之一：

   a. **OS_CLIENT_CONFIG_FILE** 环境变量的值

   b. 当前目录

   c. 特定于 Unix 的用户配置目录，如 **~/.config/openstack/clouds.yaml**

   d. 特定于 Unix 的站点配置目录，如 **/etc/openstack/clouds.yaml**
   安装程序会按照以上顺序搜索 **clouds.yaml**。

9.3.12. 创建安装配置文件

您可以自定义在 Red Hat OpenStack Platform（RHOSP）上安装的 OpenShift Container Platform 集群。

先决条件

- 获取 OpenShift Container Platform 安装程序以及集群的 pull secret。

流程

1. 创建 **install-config.yaml** 文件。
   a. 更改到包含安装程序的目录，再运行以下命令：

```
$ ./openshift-install create install-config --dir <installation_directory> 1
```

   1 对于 **<installation_directory>**，请指定用于保存安装程序所创建的文件的目录名称。

重要

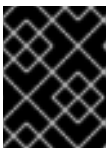
指定一个空目录。一些安装信息，如 bootstrap X.509 证书，有较短的过期间隔，因此不要重复使用安装目录。如果要重复使用另一个集群安装中的个别文件，可以将其复制到您的目录中。但是，一些安装数据的文件名可能会在发行版本之间有所改变。从 OpenShift Container Platform 老版本中复制安装文件时要格外小心。

   b. 在提示符处，提供您的云的配置详情：
      i. 可选：选择用来访问集群机器的 SSH 密钥。

注意

对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 **ssh-agent** 进程使用的 SSH 密钥。

      ii. 选择 **openstack** 作为目标平台。
      iii. 指定用于安装集群的 Red Hat OpenStack Platform (RHOSP) 外部网络名称。
      iv. 指定用于从外部访问 OpenShift API 的浮动 IP 地址。
      v. 指定至少有 16 GB RAM 用于 control plane 节点，以及计算节点的 8 GB RAM。
      vi. 选择集群要部署到的基域。所有 DNS 记录都将是这个基域的子域，并包含集群名称。
      vii. 为集群输入一个名称。名称不能多于 14 个字符。
      viii. 粘贴 Red Hat OpenShift Cluster Manager 中的 pull secret 。
2. 修改 **install-config.yaml** 文件。您可以在**安装配置参数**部分中找到有关可用参数的更多信息。
3. 备份 **install-config.yaml** 文件，以便用于安装多个集群。



重要

install-config.yaml 文件会在安装过程中消耗掉。如果要重复使用此文件，必须现在备份。

现在，文件 **install-config.yaml** 位于您指定的目录中。

9.3.13. 安装配置参数

在部署 OpenShift Container Platform 集群前，您可以提供参数值，以描述托管集群的云平台的帐户并选择性地自定义集群平台。在创建 **install-config.yaml** 安装配置文件时，您可以通过命令行来提供所需的参数的值。如果要自定义集群，可以修改 **install-config.yaml** 文件来提供关于平台的更多信息。

注意

安装之后，您无法修改 **install-config.yaml** 文件中的这些参数。

重要

openshift-install 命令不验证参数的字段名称。如果指定了不正确的名称，则不会创建相关的文件或对象，且不会报告错误。确保所有指定的参数的字段名称都正确。

9.3.13.1. 所需的配置参数

下表描述了所需的安装配置参数：

表 9.14. 所需的参数

参数	描述	值
apiVersion	**install-config.yaml** 内容的 API 版本。当前版本是 **v1**。安装程序还可能支持旧的 API 版本。	字符串
baseDomain	云供应商的基域。此基础域用于创建到 OpenShift Container Platform 集群组件的路由。集群的完整 DNS 名称是 **baseDomain** 和 **metadata.name** 参数值的组合，其格式为 **<metadata.name>.<baseDomain>**。	完全限定域名或子域名，如 **example.com**。
metadata	Kubernetes 资源 **ObjectMeta**，其中只消耗 **name** 参数。	对象
metadata.name	集群的名称。集群的 DNS 记录是 **{{.metadata.name}}.{{.baseDomain}}** 的子域。	小写字母,连字符(-)和句点(.)的字符串，如 **dev**。该字符串长度必须为 14 个字符或更少。
platform	执行安装的具体平台配置：**aws**、**baremetal**、**azure**、**openstack**、**ovirt**、**vsphere**。有关 **platform.<platform>** 参数的额外信息，请参考下表来了解您的具体平台。	对象

参数	描述	值

pullSecret	从 Red Hat OpenShift Cluster Manager 获取 pull secret，验证从 Quay.io 等服务中下载 OpenShift Container Platform 组件的容器镜像。	<pre>{
		  "auths":{
		  "cloud.openshift.com":{
		  "auth":"b3Blb=",
		  "email":"you@example.com"
		  },
		  "quay.io":{
		  "auth":"b3Blb=",
		  "email":"you@example.com"
		  }
		  }
		}</pre>

9.3.13.2. 网络配置参数

您可以根据现有网络基础架构的要求自定义安装配置。例如，您可以扩展集群网络的 IP 地址块，或者提供不同于默认值的不同 IP 地址块。

只支持 IPv4 地址。

表 9.15. 网络参数

参数	描述	值
networking	集群网络的配置。	对象 **注意** 您不能在安装后修改 **networking** 对象指定的参数。
networking.networkType	要安装的集群网络供应商 Container Network Interface（CNI）插件。	**OpenShiftSDN** 或 **OVNKubernetes**。默认值为 **OpenShiftSDN**。

networking.clusterNetwork	pod 的 IP 地址块。
	
	默认值为 **10.128.0.0/14**，主机前缀为 **/23**。
	
	如果您指定多个 IP 地址块，则块不得互相重叠。
		一个对象数组。例如：
		
		<pre>networking:
		  clusterNetwork:
		  - cidr: 10.128.0.0/14
		  hostPrefix: 23</pre>

networking.clusterNetwork.cidr	使用 **networking.clusterNetwork** 时需要此项。IP 地址块。 一个 IPv4 网络。	使用 CIDR 形式的 IP 地址块。IPv4 块的前缀长度介于 **0** 到 **32** 之间。

参数	描述	值
networking.clusterNetwork.hostPrefix	分配给每个单独节点的子网前缀长度。例如，如果 **hostPrefix** 设为 **23**，则每个节点从所给的 **cidr** 中分配一个 **/23** 子网。**hostPrefix** 值 **23** 提供 510（2^(32 - 23)- 2）个 pod IP 地址。	子网前缀。 默认值为 **23**。
networking.serviceNetwork	服务的 IP 地址块。默认值为 **172.30.0.0/16**。 OpenShift SDN 和 OVN-Kubernetes 网络供应商只支持服务网络的一个 IP 地址块。	CIDR 格式具有 IP 地址块的数组。例如： networking: serviceNetwork: - 172.30.0.0/16
networking.machineNetwork	机器的 IP 地址块。 如果您指定多个 IP 地址块，则块不得互相重叠。	一个对象数组。例如： networking: machineNetwork: - cidr: 10.0.0.0/16
networking.machineNetwork.cidr	使用 **networking.machineNetwork** 时需要。IP 地址块。libvirt 以外的所有平台的默认值为 **10.0.0.0/16**。对于 libvirt，默认值为 **192.168.126.0/24**。	CIDR 表示法中的 IP 网络块。 例如： **10.0.0.0/16**。 **注意** 将 **networking.machineNetwork** 设置为与首选 NIC 所在的 CIDR 匹配。

9.3.13.3. 可选配置参数

下表描述了可选安装配置参数：

表 9.16. 可选参数

参数	描述	值
additionalTrustBundle	添加到节点可信证书存储中的 PEM 编码 X.509 证书捆绑包。配置了代理时，也可以使用这个信任捆绑包。	字符串
compute	组成计算节点的机器的配置。	machine-pool 对象的数组。详情请查看以下"Machine-pool"表。

参数	描述	值
compute.architecture	决定池中机器的指令集合架构。目前不支持异构集群，因此所有池都必须指定相同的架构。有效值为 **amd64** （默认值）。	字符串
compute.hyperthreading	是否在计算机器上启用或禁用并发多线程或**超线程**。默认情况下，启用并发多线程以提高机器内核的性能。 **重要** 如果禁用并发多线程，请确保在容量规划时考虑到机器性能可能会显著降低的问题。	**Enabled** 或 **Disabled**
compute.name	使用 **compute** 时需要此值。机器池的名称。	**worker**
compute.platform	使用 **compute** 时需要此值。使用此参数指定托管 worker 机器的云供应商。此参数值必须与 **controlPlane.platform** 参数值匹配。	**aws**、**azure**、**gcp**、**openstack**、**ovirt**、**vsphere** 或 **{}**
compute.replicas	要置备的计算机器数量，也称为 worker 机器。	大于或等于 **2** 的正整数。默认值为 **3**。
controlPlane	组成 control plane 的机器的配置。	**MachinePool** 对象的数组。详情请查看以下"Machine-pool"表。
controlPlane.architecture	决定池中机器的指令集合架构。目前不支持异构集群，因此所有池都必须指定相同的架构。有效值为 **amd64** （默认值）。	字符串
controlPlane.hyperthreading	是否在 control plane 机器上启用或禁用并发多线程或**超线程**。默认情况下，启用并发多线程以提高机器内核的性能。 **重要** 如果禁用并发多线程，请确保在容量规划时考虑到机器性能可能会显著降低的问题。	**Enabled** 或 **Disabled**

参数	描述	值
controlPlane.name	使用 **controlPlane** 时需要。机器池的名称。	**master**
controlPlane.platform	使用 **controlPlane** 时需要。使用此参数指定托管 control plane 机器的云供应商。此参数值必须与 **compute.platform** 参数值匹配。	**aws**、**azure**、**gcp**、**openstack**、**ovirt**、**vsphere** 或 **{}**
controlPlane.replicas	要置备的 control plane 机器数量。	唯一支持的值是 **3**，它是默认值。
credentialsMode	Cloud Credential Operator（CCO）模式。如果没有指定任何模式，CCO 会动态地尝试决定提供的凭证的功能，在支持多个模式的平台上使用 mint 模式。  **注意** 不是所有 CCO 模式都支持所有云供应商。如需有关 CCO 模式的更多信息，请参阅*Red Hat Operator 参考指南*内容中的 *Cloud Credential Operator* 条目。	**Mint**、**Passthrough**、**Manual** 或空字符串(**""**)。
fips	启用或禁用 FIPS 模式。默认为 **false**（禁用）。如果启用了 FIPS 模式，运行 OpenShift Container Platform 的 Red Hat Enterprise Linux CoreOS (RHCOS) 机器会绕过默认的 Kubernetes 加密套件，并使用由 RHCOS 提供的加密模块。  **重要** 只有在 **x86_64** 架构中的 OpenShift Container Platform 部署支持 FIPS 验证的/Modules in Process 加密库。  **注意** 如果使用 Azure File 存储，则无法启用 FIPS 模式。	**false** 或 **true**

参数	描述	值
imageContentSources	release-image 内容的源和仓库。	对象数组。包括一个 **source** 以及可选的 **mirrors**，如下表所示。
imageContentSources.source	使用 **imageContentSources** 时需要。指定用户在镜像拉取规格中引用的仓库。	字符串
imageContentSources.mirrors	指定可能还包含同一镜像的一个或多个仓库。	字符串数组
publish	如何发布或公开集群的面向用户的端点，如 Kubernetes API、OpenShift 路由。	**Internal** 或 **External**。默认值为 **External**。 在非云平台上不支持将此字段设置为 **Internal**。 **重要** 如果将字段的值设为 **Internal**，集群将无法运行。如需更多信息，请参阅 BZ#1953035。
sshKey	用于验证集群机器访问的 SSH 密钥或密钥。 **注意** 对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 **ssh-agent** 进程使用的 SSH 密钥。	一个或多个密钥。例如： sshKey: <key1> <key2> <key3>

9.3.13.4. 其他 Red Hat OpenStack Platform（RHOSP）配置参数

下表描述了其他 RHOSP 配置参数：

表 9.17. 其他 RHOSP 参数

参数	描述	值
compute.platform.openstack.rootVolume.size	对于计算机器，以 GB 为单位表示的根卷大小。如果您不设置这个值，机器将使用临时存储。	整数，如 **30**。
compute.platform.openstack.rootVolume.type	对于计算机器，根卷的类型。	字符串，如 **performance**。
controlPlane.platform.openstack.rootVolume.size	对于 control plane 机器，以 GB 为单位表示的根卷大小。如果您不设置这个值，机器将使用临时存储。	整数，如 **30**。
controlPlane.platform.openstack.rootVolume.type	对于 control plane 机器，根卷的类型。	字符串，如 **performance**。
platform.openstack.cloud	要使用的 RHOSP 云的名称，来自于 **clouds.yaml** 文件中的云列表。	字符串，如 **MyCloud**。
platform.openstack.externalNetwork	用于安装的 RHOSP 外部网络名称。	字符串，如 **external**。
platform.openstack.computeFlavor	用于 control plane 和计算机器的 RHOSP 类别。	字符串，如 **m1.xlarge**。

9.3.13.5. 可选 RHOSP 配置参数

下表描述了可选 RHOSP 配置参数：

表 9.18. 可选的 RHOSP 参数

参数	描述	值
compute.platform.openstack.additionalNetworkIDs	与计算机器关联的其他网络。不能为额外网络创建允许的地址对。	一个或多个 UUID 列表作为字符串。例如：**fa806b2f-ac49-4bce-b9db-124bc64209bf**。
compute.platform.openstack.additionalSecurityGroupIDs	与计算机器关联的其他安全组。	一个或多个 UUID 列表作为字符串。例如：**7ee219f3-d2e9-48a1-96c2-e7429f1b0da7**。

参数	描述	值
compute.platform.openstack.zones	RHOSP Compute（Nova）可用区（AZ）在其中安装机器。如果没有设置此参数，安装程序会依赖于配置了 RHOSP 管理员的 Nova 的默认设置。 在使用 Kuryr 的集群上，RHOSP Octavia 不支持可用域。负载均衡器，如果您使用 Amphora 供应商驱动程序，则依赖 Amphora 虚拟机的 OpenShift Container Platform 服务不会根据此属性的值创建。	字符串列表。例如：**["zone-1", "zone-2"]**。
controlPlane.platform.openstack.additionalNetworkIDs	与 control plane 机器关联的额外网络。不能为额外网络创建允许的地址对。	一个或多个 UUID 列表作为字符串。例如：**fa806b2f-ac49-4bce-b9db-124bc64209bf**。
controlPlane.platform.openstack.additionalSecurityGroupIDs	与 control plane 机器关联的其他安全组。	一个或多个 UUID 列表作为字符串。例如：**7ee219f3-d2e9-48a1-96c2-e7429f1b0da7**。
controlPlane.platform.openstack.zones	RHOSP Compute（Nova）可用区（AZ）在其中安装机器。如果没有设置此参数，安装程序会依赖于配置了 RHOSP 管理员的 Nova 的默认设置。 在使用 Kuryr 的集群上，RHOSP Octavia 不支持可用域。负载均衡器，如果您使用 Amphora 供应商驱动程序，则依赖 Amphora 虚拟机的 OpenShift Container Platform 服务不会根据此属性的值创建。	字符串列表。例如：**["zone-1", "zone-2"]**。
platform.openstack.clusterOSImage	安装程序从中下载 RHCOS 镜像的位置。 您必须设置此参数以便在受限网络中执行安装。	HTTP 或 HTTPS URL，可选使用 SHA-256 checksum。 例如：**http://mirror.example.com/images/rhcos-43.81.201912131630.0-openstack.x86_64.qcow2.gz?sha256=ffebbd68e8a1f2a245ca19522c16c86f67f9ac8e4e0c1f0a812b068b16f7265d**。该值也可以是现有 Glance 镜像的名称，如 **my-rhcos**。

参数	描述	值

platform.openstack.defaultMachinePlatform	默认机器池平台配置。	<pre>{
		  "type": "ml.large",
		  "rootVolume": {
		  "size": 30,
		  "type": "performance"
		  }
		}</pre>

platform.openstack.ingressFloatingIP	与 Ingress 端口关联的现有浮动 IP 地址。要使用此属性，还必须定义 **platform.openstack.externalNetwork** 属性。	IP 地址，如 **128.0.0.1**。
platform.openstack.lbFloatingIP	与 API 负载均衡器关联的现有浮动 IP 地址。要使用此属性，还必须定义 **platform.openstack.externalNetwork** 属性。	IP 地址，如 **128.0.0.1**。
platform.openstack.externalDNS	集群实例用于进行 DNS 解析的外部 DNS 服务器的 IP 地址。	一个 IP 地址列表作为字符串。例如，**["8.8.8.8", "192.168.1.12"]**。
platform.openstack.machinesSubnet	集群节点使用的 RHOSP 子网的 UUID。在这个子网上创建节点和虚拟 IP（VIP）端口。 **networking.machineNetwork** 中的第一个项需要和 **machinesSubnet** 的值匹配。 如果部署到自定义子网中，则无法将外部 DNS 服务器指定到 OpenShift Container Platform 安装程序。反之，把 DNS 添加到 RHOSP 的子网。	作为字符串的 UUID。例如： **fa806b2f-ac49-4bce-b9db-124bc64209bf**。

9.3.13.6. RHOSP 部署中的自定义子网

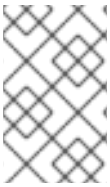
另外，您还可以在您选择的 Red Hat OpenStack Platform（RHOSP）子网中部署集群。子网的 GUID 作为 **install-config.yaml** 文件中的 **platform.openstack.machinesSubnet** 的值传递。

此子网被用作集群的主子网，在其上创建节点和端口。

在使用自定义子网运行 OpenShift Container Platform 安装程序前，请验证：

- 目标网络和子网可用。

- 目标子网上启用了 DHCP。
- 您可提供在目标网络上有创建端口权限的安装程序凭证。
- 如果您的网络配置需要一个路由器，它会在 RHOSP 中创建。有些配置依赖于路由器来转换浮动 IP 地址。
- 您的网络配置不依赖于供应商网络。不支持提供商网络。



注意

默认情况下，API VIP 使用 x.x.x.5，Ingress VIP 从网络 CIDR 块获取 x.x.x.7。要覆盖这些默认值，为 DHCP 分配池以外的 **platform.openstack.apiVIP** 和 **platform.openstack.ingressVIP** 设置值。

9.3.13.7. RHOSP 的自定义 install-config.yaml 文件示例

此示例 **install-config.yaml** 展示了所有可能的 Red Hat OpenStack Platform (RHOSP) 自定义选项。

重要

此示例文件仅供参考。您必须使用安装程序来获取 **install-config.yaml** 文件。

```
apiVersion: v1
baseDomain: example.com
clusterID: os-test
controlPlane:
  name: master
  platform: {}
  replicas: 3
compute:
- name: worker
  platform:
    openstack:
      type: ml.large
  replicas: 3
metadata:
  name: example
networking:
  clusterNetwork:
  - cidr: 10.128.0.0/14
    hostPrefix: 23
  machineNetwork:
  - cidr: 10.0.0.0/16
  serviceNetwork:
  - 172.30.0.0/16
  networkType: OpenShiftSDN
platform:
  openstack:
    cloud: mycloud
    externalNetwork: external
    computeFlavor: m1.xlarge
    lbFloatingIP: 128.0.0.1
```

```
fips: false
pullSecret: '{"auths": ...}'
sshKey: ssh-ed25519 AAAA...
```

9.3.13.8. 为机器设置自定义子网

安装程序默认使用的 IP 范围可能与您在安装 OpenShift Container Platform 时创建的 Neutron 子网不匹配。如有必要，通过编辑安装配置文件来更新新机器的 CIDR 值。

先决条件

- 有 OpenShift Container Platform 安装程序生成的 **install-config.yaml** 文件。

流程

1. 在命令行中进入包含 **install-config.yaml** 的目录。
2. 在该目录中，运行脚本来编辑 **install-config.yaml** 文件或手动更新该文件：
   - 要使用脚本设置值，请运行：

```
$ python -c '
import yaml;
path = "install-config.yaml";
data = yaml.safe_load(open(path));
data["networking"]["machineNetwork"] = [{"cidr": "192.168.0.0/18"}]; 1
open(path, "w").write(yaml.dump(data, default_flow_style=False))'
```

1 插入一个与您指定的 Neutron 子网匹配的值，如 **192.0.2.0/24**。

   - 要手动设置这个值，请打开该文件并将 **networking.machineCIDR** 的值设置为与您预期的 Neutron 子网匹配的内容。

9.3.13.9. 清空计算机器池

要进行使用您自己的基础架构的安装，请将安装配置文件中的计算机器数量设置为零。之后，您可以手动创建这些机器。

先决条件

- 有 OpenShift Container Platform 安装程序生成的 **install-config.yaml** 文件。

流程

1. 在命令行中进入包含 **install-config.yaml** 的目录。
2. 在该目录中，运行脚本来编辑 **install-config.yaml** 文件或手动更新该文件：
   - 要使用脚本设置值，请运行：

```
$ python -c '
import yaml;
path = "install-config.yaml";
```

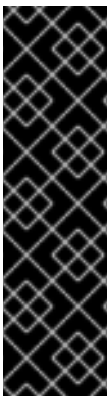
```
data = yaml.safe_load(open(path));
data["compute"][0]["replicas"] = 0;
open(path, "w").write(yaml.dump(data, default_flow_style=False))'
```

- 要手动设置值，打开文件并将 **compute.<first entry>.replicas** 的值设置为 **0**。

9.3.14. 创建 Kubernetes 清单和 Ignition 配置文件

由于您必须修改一些集群定义文件并要手动启动集群机器，因此您必须生成 Kubernetes 清单和 Ignition 配置文件，集群需要这两项来创建其机器。

安装配置文件转换为 Kubernetes 清单。清单嵌套到 Ignition 配置文件中，稍后用于创建集群。



重要

- 安装程序生成的 Ignition 配置文件包含在 24 小时后过期的证书，然后在过期时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外情况是，您需要手动批准待处理的 **node-bootstrapper** 证书签名请求（CSR）来恢复 kubelet 证书。如需更多信息，请参阅*从过期的 control plane 证书中恢复*的文档。
- 建议您在生成 12 小时后使用 Ignition 配置文件，因为集群安装后 24 小时证书从 16 小时轮转至 22 小时。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中运行证书更新时避免安装失败。

先决条件

- 已获得 OpenShift Container Platform 安装程序。
- 已创建 **install-config.yaml** 安装配置文件。

流程

1. 切换到包含安装程序的目录，并为集群生成 Kubernetes 清单：

```
$ ./openshift-install create manifests --dir <installation_directory> 1
```

1 对于 **<installation_directory>**，请指定含有您创建的 **install-config.yaml** 文件的安装目录。

2. 删除定义 control plane 机器的 Kubernetes 清单文件以及计算机器集：

```
$ rm -f openshift/99_openshift-cluster-api_master-machines-*.yaml openshift/99_openshift-cluster-api_worker-machineset-*.yaml
```

由于您要自行创建和管理这些资源，因此不必初始化这些资源。

- 您可以使用机器 API 来保留机器集文件来创建计算机器，但您必须更新对其的引用，以匹配您的环境。

3. 检查 **<installation_directory>/manifests/cluster-scheduler-02-config.yml** Kubernetes 清单文件中的 **mastersSchedulable** 参数是否已设置为 **false**。此设置可防止在 control plane 机器上调度 pod:

a. 打开 **<installation_directory>/manifests/cluster-scheduler-02-config.yml** 文件。

b. 找到 **mastersSchedulable** 参数并确保它被设置为 **false**。

c. 保存并退出文件。

4. 要创建 Ignition 配置文件，从包含安装程序的目录运行以下命令：

```
$ ./openshift-install create ignition-configs --dir <installation_directory> 1
```

1 对于 **<installation_directory>**，请指定相同的安装目录。

该目录中将生成以下文件：

```
.
├── auth
│   ├── kubeadmin-password
│   └── kubeconfig
├── bootstrap.ign
├── master.ign
├── metadata.json
└── worker.ign
```

5. 将元数据文件的 **infraID** 键导出为环境变量：

```
$ export INFRA_ID=$(jq -r .infraID metadata.json)
```

提示

从 **metadata.json** 中提取 **infraID** 键，并将其用作您创建的所有 RHOSP 资源的前缀。通过这样做，您可以避免在同一项目中进行多个部署时的名称冲突。

9.3.15. 准备 bootstrap Ignition 文件

OpenShift Container Platform 安装过程依赖于从 bootstrap Ignition 配置文件创建的 bootstrap 机器。

编辑该文件并上传该文件。然后，创建 Red Hat OpenStack Platform（RHOSP）用来下载主文件的辅助 bootstrap Ignition 配置文件。

先决条件

- 您有安装程序生成的 bootstrap Ignition 文件，即 **bootstrap.ign**。
- 安装程序元数据文件中的基础架构 ID 被设置为环境变量（**$INFRA_ID**）。
  - 如果未设置变量，请参阅 **创建 Kubernetes 清单和 Ignition 配置文件**。
- 可以使用 HTTP(S) 来存储 bootstrap ignition 文件。
  - 所记录的步骤使用 RHOSP 镜像服务（Glance），但也可以使用 RHOSP Storage 服务（Swift）、Amazon S3、内部 HTTP 服务器或临时 Nova 服务器。

流程

1. 运行以下 Python 脚本。该脚本修改 bootstrap Ignition 文件，以设置主机名，并在运行时设置 CA 证书文件：

```
import base64
import json
import os

with open('bootstrap.ign', 'r') as f:
    ignition = json.load(f)

files = ignition['storage'].get('files', [])

infra_id = os.environ.get('INFRA_ID', 'openshift').encode()
hostname_b64 = base64.standard_b64encode(infra_id + b'-bootstrap\n').decode().strip()
files.append(
{
    'path': '/etc/hostname',
    'mode': 420,
    'contents': {
        'source': 'data:text/plain;charset=utf-8;base64,' + hostname_b64
    }
})

ca_cert_path = os.environ.get('OS_CACERT', '')
if ca_cert_path:
    with open(ca_cert_path, 'r') as f:
        ca_cert = f.read().encode()
        ca_cert_b64 = base64.standard_b64encode(ca_cert).decode().strip()

    files.append(
    {
        'path': '/opt/openshift/tls/cloud-ca-cert.pem',
        'mode': 420,
        'contents': {
            'source': 'data:text/plain;charset=utf-8;base64,' + ca_cert_b64
        }
    })

ignition['storage']['files'] = files;

with open('bootstrap.ign', 'w') as f:
    json.dump(ignition, f)
```

2. 使用 RHOSP CLI，创建使用 bootstrap Ignition 文件的镜像：

```
$ openstack image create --disk-format=raw --container-format=bare --file bootstrap.ign
<image_name>
```

3. 获取镜像的详情：

```
$ openstack image show <image_name>
```

请记录 **file** 值 ; 它需要遵循 **v2/images/<image_ID>/file** 格式。

注意

验证您创建的镜像是否活跃。

4. 检索镜像服务的公共地址：

```
$ openstack catalog show image
```

5. 将公共地址与镜像的 **file** 值合并，并在存储位置保存结果。位置遵循 **<image_service_public_URL>/v2/images/<image_ID>/file** 格式。

6. 生成身份验证令牌并保存令牌 ID:

```
$ openstack token issue -c id -f value
```

7. 将以下内容插入到名为 **$INFRA_ID-bootstrap-ignition.json** 的文件中，并编辑位置拥有者以匹配您自己的值：

```
{
  "ignition": {
    "config": {
      "merge": [{
        "source": "<storage_url>", 1
        "httpHeaders": [{
          "name": "X-Auth-Token", 2
          "value": "<token_ID>" 3
        }]
      }]
    },
    "security": {
      "tls": {
        "certificateAuthorities": [{
          "source": "data:text/plain;charset=utf-8;base64,<base64_encoded_certificate>" 4
        }]
      }
    },
    "version": "3.1.0"
  }
}
```

1. 将 **ignition.config.merge.source** 的值替换为 bootstrap Ignition 文件存储 URL。
2. 在 **httpHeaders** 中将 **name** 设置为 **"X-Auth-Token"**。
3. 在 **httpHeaders** 中将 **value** 设为您的令牌 ID。
4. 如果 bootstrap Ignition 文件服务器使用自签名证书,请包括以 base64 编码的证书。

8. 保存二级 Ignition 配置文件。

bootstrap Ignition 数据将在安装过程中传递给 RHOSP。

警告

bootstrap Ignition 文件包含敏感信息，如 **clouds.yaml** 凭证。确定您将其保存在安全的地方，并在完成安装后将其删除。

9.3.16. 在 RHOSP 上创建 control plane Ignition 配置文件

在您自己的基础架构的 Red Hat OpenStack Platform（RHOSP）上安装 OpenShift Container Platform 需要 control plane Ignition 配置文件。您必须创建多个配置文件。

注意

与 bootstrap Ignition 配置一样，您必须明确为每个 control plane 机器定义主机名。

先决条件

- 来自安装程序元数据文件中的基础架构 ID 被设置为环境变量（**$INFRA_ID**）。
  - 如果未设置变量，请参阅“创建 Kubernetes 清单和 Ignition 配置文件”。

流程

- 在命令行中运行以下 Python 脚本：

```
$ for index in $(seq 0 2); do
    MASTER_HOSTNAME="$INFRA_ID-master-$index\n"
    python -c "import base64, json, sys;
ignition = json.load(sys.stdin);
storage = ignition.get('storage', {});
files = storage.get('files', []);
files.append({'path': '/etc/hostname', 'mode': 420, 'contents': {'source':
'data:text/plain;charset=utf-8;base64,' +
base64.standard_b64encode(b'$MASTER_HOSTNAME').decode().strip(), 'verification': {}},
'filesystem': 'root'});
storage['files'] = files;
ignition['storage'] = storage
json.dump(ignition, sys.stdout)" <master.ign >"$INFRA_ID-master-$index-ignition.json"
done
```

  您现在有三个 control plane Ignition 文件：**<INFRA_ID>-master-0-ignition.json**、**<INFRA_ID>-master-1-ignition.json** 和 **<INFRA_ID>-master-2-ignition.json**。

9.3.17. 在 RHOSP 上创建网络资源

在您自己的基础架构的 Red Hat OpenStack Platform（RHOSP）安装上创建 OpenShift Container Platform 所需的网络资源。为节省时间，可以运行提供的 Ansible playbook 来生成安全组、网络、子网、路由器和端口。

先决条件

- Python 3 已安装在您的机器上。
- 您下载了"下载 playbook 依赖项"中的模块。
- 下载了"下载安装 playbook"中的 playbook。

流程

1. 可选：为 **inventory.yaml** playbook 添加一个外部网络值：

   inventory.yaml Ansible playbook 中的外部网络值示例

```
...
    # The public network providing connectivity to the cluster. If not
    # provided, the cluster external connectivity must be provided in another
    # way.

    # Required for os_api_fip, os_ingress_fip, os_bootstrap_fip.
    os_external_network: 'external'
...
```

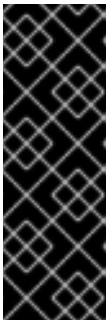
重要

如果没有为 **inventory.yaml** 文件中的 **os_external_network** 提供值，则必须确保虚拟机可以自行访问 Glance 和外部连接。

2. 可选：将外部网络和浮动 IP（FIP）地址值添加到 **inventory.yaml** playbook：

   inventory.yaml Ansible playbook 中的 FIP 值示例

```
...
    # OpenShift API floating IP address. If this value is non-empty, the
    # corresponding floating IP will be attached to the Control Plane to
    # serve the OpenShift API.
    os_api_fip: '203.0.113.23'

    # OpenShift Ingress floating IP address. If this value is non-empty, the
    # corresponding floating IP will be attached to the worker nodes to serve
    # the applications.
    os_ingress_fip: '203.0.113.19'

    # If this value is non-empty, the corresponding floating IP will be
    # attached to the bootstrap machine. This is needed for collecting logs
    # in case of install failure.
    os_bootstrap_fip: '203.0.113.20'
```



重要

如果您没有为 **os_api_fip** 和 **os_ingress_fip** 定义值，则必须执行安装后的网络配置。

如果您没有为 **os_bootstrap_fip** 定义值，安装程序将无法从失败的安装中下载调试信息。

如需更多信息，请参阅"启用对环境的访问"。

3. 在命令行中，通过运行 **security-groups.yaml** playbook 来创建安全组：

```
$ ansible-playbook -i inventory.yaml security-groups.yaml
```

4. 在命令行中，通过运行 **network.yaml** playbook 来创建一个网络、子网和路由器：

```
$ ansible-playbook -i inventory.yaml network.yaml
```

5. 可选：如果要控制 Nova 服务器使用的默认解析程序，请运行 RHOSP CLI 命令：

```
$ openstack subnet set --dns-nameserver <server_1> --dns-nameserver <server_2>
"$INFRA_ID-nodes"
```

9.3.18. 在 RHOSP 上创建 bootstrap 机器

创建 bootstrap 机器，为其提供在 Red Hat OpenStack Platform（RHOSP）上运行所需的网络访问权限。红帽提供了一个 Ansible playbook，您可运行它来简化此过程。

先决条件

- 您下载了"下载 playbook 依赖项"中的模块。
- 下载了"下载安装 playbook"中的 playbook。
- **inventory.yaml**、**common.yaml** 和 **bootstrap.yaml** Ansible playbook 位于一个通用目录中。
- 安装程序创建的 **metadata.json** 文件与 Ansible playbook 位于同一个目录中。

流程

1. 在命令行中，将工作目录改为 playbook 的位置。
2. 在命令行中运行 **bootstrap.yaml** playbook：

```
$ ansible-playbook -i inventory.yaml bootstrap.yaml
```

3. bootstrap 服务器可用后，查看日志以验证是否收到 Ignition 文件：

```
$ openstack console log show "$INFRA_ID-bootstrap"
```

9.3.19. 在 RHOSP 中创建 control plane 机器

使用您生成的 Ignition 配置文件创建三台 control plane 机器。红帽提供了一个 Ansible playbook，您可运行它来简化此过程。

先决条件

- 您下载了"下载 playbook 依赖项"中的模块。
- 下载了"下载安装 playbook"中的 playbook。
- 来自安装程序元数据文件中的基础架构 ID 被设置为环境变量（**$INFRA_ID**）。
- **inventory.yaml**、**common.yaml** 和 **control-plane.yaml** Ansible playbook 位于一个通用目录中。
- 您有三个在"Creating control plane Ignition 配置文件"中创建的 Ignition 文件。

流程

1. 在命令行中，将工作目录改为 playbook 的位置。
2. 如果 control plane Ignition 配置文件尚未位于工作目录中，将其复制到其中。
3. 在命令行中运行 **control-plane.yaml** playbook：

```
$ ansible-playbook -i inventory.yaml control-plane.yaml
```

4. 运行以下命令来监控 bootstrap 过程：

```
$ openshift-install wait-for bootstrap-complete
```

   您会看到确认 control plane 机器正在运行并加入集群的消息：

```
INFO API v1.14.6+f9b5405 up
INFO Waiting up to 30m0s for bootstrapping to complete...
...
INFO It is now safe to remove the bootstrap resources
```

9.3.20. 使用 CLI 登录到集群

您可以通过导出集群 **kubeconfig** 文件，以默认系统用户身份登录集群。**kubeconfig** 文件包含关于集群的信息，供 CLI 用于将客户端连接到正确集群和 API 服务器。该文件特只适用于一个特定的集群，在 OpenShift Container Platform 安装过程中创建。

先决条件

- 已部署了 OpenShift Container Platform 集群。
- 已安装 **oc** CLI。

流程

1. 导出 **kubeadmin** 凭证：

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

2. 使用导出的配置，验证能否成功运行 **oc** 命令：

```
$ oc whoami
```

输出示例

```
system:admin
```

9.3.21. 从 RHOSP 删除 bootstrap 资源

删除您不再需要的 bootstrap 资源。

先决条件

- 您下载了"下载 playbook 依赖项"中的模块。
- 下载了"下载安装 playbook"中的 playbook。
- **inventory.yaml**、**common.yaml** 和 **down-bootstrap.yaml** Ansible playbook 位于一个通用目录中。
- control plane 机器正在运行。
  - 如果您不知道机器的状态，请参阅"验证集群状态"。

流程

1. 在命令行中，将工作目录改为 playbook 的位置。
2. 在命令行中运行 **down-bootstrap.yaml** playbook：

```
$ ansible-playbook -i inventory.yaml down-bootstrap.yaml
```

bootstrap 端口、服务器和浮动 IP 地址会被删除。

警告

如果您之前没有禁用 bootstrap Ignition 文件 URL，现在需要禁用。

9.3.22. 在 RHOSP 上创建计算机器

启动 control plane 后，创建计算机器。红帽提供了一个 Ansible playbook，您可运行它来简化此过程。

先决条件

- 您下载了"下载 playbook 依赖项"中的模块。

- 下载了"下载安装 playbook"中的 playbook。
- **inventory.yaml**、**common.yaml** 和 **compute-nodes.yaml** Ansible playbook 位于一个通用目录中。
- 安装程序创建的 **metadata.json** 文件与 Ansible playbook 位于同一个目录中。
- control plane 处于活跃状态。

流程

1. 在命令行中，将工作目录改为 playbook 的位置。
2. 在命令行中运行 playbook:

```
$ ansible-playbook -i inventory.yaml compute-nodes.yaml
```

后续步骤

- 批准机器的证书签名请求。

9.3.23. 批准机器的证书签名请求

将机器添加到集群时，会为您添加的每台机器生成两个待处理证书签名请求（CSR）。您必须确认这些 CSR 已获得批准，或根据需要自行批准。客户端请求必须首先被批准，然后是服务器请求。

先决条件

- 您已将机器添加到集群中。

流程

1. 确认集群可以识别这些机器：

```
$ oc get nodes
```

输出示例

```
NAME     STATUS    ROLES   AGE  VERSION
master-0  Ready     master  63m  v1.19.0
master-1  Ready     master  63m  v1.19.0
master-2  Ready     master  64m  v1.19.0
```

输出将列出您创建的所有机器。

注意

在一些 CSR 被批准前，以上输出可能不包括计算节点（也称为 worker 节点）。

2. 检查待处理的 CSR，并确保可以看到添加到集群中的每台机器都有 **Pending** 或 **Approved** 状态的客户端请求：

```
$ oc get csr
```

-

输出示例

```
NAME        AGE    REQUESTOR                                                    CONDITION
csr-8b2br   15m    system:serviceaccount:openshift-machine-config-operator:node-
bootstrapper   Pending
csr-8vnps   15m    system:serviceaccount:openshift-machine-config-operator:node-
bootstrapper   Pending
...
```

在本例中，两台机器加入了集群。您可能会在列表中看到更多已批准的 CSR。

3. 如果 CSR 没有获得批准，请在所添加机器的所有待处理 CSR 都处于 **Pending** 状态后，为您的集群机器批准这些 CSR：

注意

由于 CSR 会自动轮转，因此请在将机器添加到集群后一小时内批准您的 CSR。如果没有在一小时内批准，证书将会轮转，每个节点将会存在多个证书。您必须批准所有这些证书。批准客户端 CSR 后， Kubelet 为服务证书创建辅助 CSR，这需要手动批准。然后，如果 Kubelet 请求具有相同参数的新证书，则 **machine-approver** 会自动批准后续服务证书续订请求。

注意

对于在未启用机器 API 的平台中运行的集群，如裸机和其他用户置备的基础架构，必须采用一种方法自动批准 kubelet 提供证书请求（CSR）。如果没有批准请求，则 **oc exec**、**oc rsh** 和 **oc logs** 命令将无法成功，因为 API 服务器连接到 kubelet 时需要服务证书。与 Kubelet 端点联系的任何操作都需要此证书批准。这个方法必须监视新的 CSR，确认 CSR 由 **system:node** 或 **system:admin** 组中的 **node-bootstrapper** 服务帐户提交，并确认节点的身份。

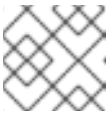
- 若要单独批准，请对每个有效的 CSR 运行以下命令：

```
$ oc adm certificate approve <csr_name> 1
```

1 **<csr_name>** 是当前 CSR 列表中 CSR 的名称。

- 要批准所有待处理的 CSR，请运行以下命令：

```
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}{{"\n"}}
{{end}}{{end}}' | xargs --no-run-if-empty oc adm certificate approve
```



注意

在有些 CSR 被批准前，一些 Operator 可能无法使用。

4. 现在，您的客户端请求已被批准，您必须查看添加到集群中的每台机器的服务器请求：

```
$ oc get csr
```

输出示例

```
NAME        AGE     REQUESTOR                                                   CONDITION
csr-bfd72   5m26s   system:node:ip-10-0-50-126.us-east-2.compute.internal       Pending
csr-c57lv   5m26s   system:node:ip-10-0-95-157.us-east-2.compute.internal       Pending
...
```

5. 如果剩余的 CSR 没有被批准，且处于 **Pending** 状态，请批准集群机器的 CSR：

- 若要单独批准，请对每个有效的 CSR 运行以下命令：

```
$ oc adm certificate approve <csr_name> 1
```

1 **<csr_name>** 是当前 CSR 列表中 CSR 的名称。

- 要批准所有待处理的 CSR，请运行以下命令：

```
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}{{"\n"}}
{{end}}{{end}}' | xargs oc adm certificate approve
```

6. 批准所有客户端和服务器 CSR 后，器将处于 **Ready** 状态。运行以下命令验证：

```
$ oc get nodes
```

输出示例

```
NAME      STATUS    ROLES   AGE  VERSION
master-0  Ready     master  73m  v1.20.0
master-1  Ready     master  73m  v1.20.0
master-2  Ready     master  74m  v1.20.0
worker-0  Ready     worker  11m  v1.20.0
worker-1  Ready     worker  11m  v1.20.0
```

注意

批准服务器 CSR 后可能需要几分钟时间让机器转换为 **Ready** 状态。

其他信息

- 如需有关 CSR 的更多信息，请参阅证书签名请求。

9.3.24. 验证安装是否成功

验证 OpenShift Container Platform 安装已完成。

先决条件

- 有安装程序（**openshift-install**）

流程

- 在命令行中运行：

```
$ openshift-install --log-level debug wait-for install-complete
```

程序输出控制台 URL 以及管理员的登录信息。

9.3.25. OpenShift Container Platform 的 Telemetry 访问

在 OpenShift Container Platform 4.6 中，默认运行的 Telemetry 服务提供有关集群健康状况和成功更新的指标，需要访问互联网。如果您的集群连接到互联网，Telemetry 会自动运行，而且集群会注册到 OpenShift Cluster Manager。

确认 OpenShift Cluster Manager 清单正确后，可以由 Telemetry 自动维护，也可以使用 OpenShift Cluster Manager 手动维护，使用订阅监控来跟踪帐户或多集群级别的 OpenShift Container Platform 订阅。

其他资源

- 有关 Telemetry 服务的更多信息，请参阅关于远程健康监控。

9.3.26. 后续步骤

- 自定义集群。
- 如果需要，您可以选择不使用远程健康报告。
- 如果您需要启用对节点端口的外部访问，请使用节点端口配置集群流量。
- 如果您没有将 RHOSP 配置为使用浮动 IP 地址接受应用程序流量，使用浮动 IP 地址配置 RHOSP 访问。

9.4. 在您自己的基础架构上带有 KURYR 的 OPENSTACK 上安装集群

在 OpenShift Container Platform 版本 4.6 中，您可以在运行于用户置备的基础架构上的 Red Hat OpenStack Platform（RHOSP）上安装集群。

通过利用您自己的基础架构，您可以将集群与现有的基础架构进行集成。和安装程序置备的安装方式相比，这个过程需要用户进行更多操作，因为您必须创建所有 RHOSP 资源，如 Nova 服务器、Neutron 端口和安全组。红帽提供了 Ansible playbook 来帮助您完成部署过程。

9.4.1. 先决条件

- 查看有关 OpenShift Container Platform 安装和更新流程的详细信息。
  - 在 *Available platforms* 部分验证 OpenShift Container Platform 4.6 是否与您的 RHOSP 版本兼容。您还可以查看 OpenShift Container Platform 在 RHOSP 中的支持 来比较不同版本的平台支持。
- 验证您的网络配置不依赖于供应商网络。不支持提供商网络。
- 具有要安装 OpenShift Container Platform 的 RHOSP 帐户
- 在您运行安装程序的机器中，有：

- 用来保存在安装过程中创建的文件的一个单一目录
- Python 3

9.4.2. 关于 Kuryr SDN

Kuryr 是一个容器网络接口（CNI）插件解决方案，它使用 Neutron 和 Octavia Red Hat OpenStack Platform（RHOSP）服务来为 pod 和服务提供网络。

Kuryr 和 OpenShift Container Platform 的集成主要针对在 RHOSP VM 上运行的 OpenShift Container Platform 集群设计。Kuryr 通过将 OpenShift Container Platform pod 插入到 RHOSP SDN 来提高网络性能。另外，它还提供 pod 和 RHOSP 虚拟实例间的互联性。

Kuryr 组件作为 pod 在 OpenShift Container Platform 中安装，使用 **openshift-kuryr** 命名空间：

- **kuryr-controller** - 在一个 **master** 节点上安装的单个服务实例。这在 OpenShift Container Platform 中建模为一个 **Deployment** 对象。
- **kuryr-cni** - 在每个 OpenShift Container Platform 节点上安装并配置 Kuryr 作为 CNI 驱动的容器。这在 OpenShift Container Platform 中建模为一个 **DaemonSet** 对象。

Kuryr 控制器监控 OpenShift Container Platform API 服务器中的 pod、服务和命名空间创建、更新和删除事件。它将 OpenShift Container Platform API 调用映射到 Neutron 和 Octavia 中的对应对象。这意味着，实现了 Neutron 中继端口功能的每个网络解决方案都可以通过 Kuryr 支持 OpenShift Container Platform。这包括开源解决方案，比如 Open vSwitch (OVS) 和 Open Virtual Network (OVN)，以及 Neutron 兼容的商业 SDN。

建议在封装的 RHOSP 租户网络上部署 OpenShift Container Platform 时使用 Kuryr，以避免出现重复封装，例如通过 RHOSP 网络运行封装的 OpenShift Container Platform SDN。

如果您使用供应商网络或租户 VLAN，则不需要使用 Kuryr 来避免重复封装。虽然性能上的优势微不足道，但根据您的配置，使用 Kuryr 避免两个覆盖可能仍然有用。

在完足以下所有条件的部署中不建议使用 Kuryr：

- RHOSP 版本早于 16
- 部署使用 UDP 服务，或者在几个 hypervisor 上使用大量 TCP 服务。

或

- **ovn-octavia** Octavia 驱动被禁用。
- 部署在几个 hypervisor 中使用了大量的 TCP 服务。

9.4.3. 在带有 Kuryr 的 OpenStack 上安装 OpenShift Container Platform 的资源指南

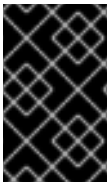
当使用 Kuryr SDN 时，pod、服务、命名空间和网络策略会使用来自 RHOSP 配额的资源，这会增加最低要求。除了默认安装需要满足的要求，Kuryr 还有一些额外的要求。

使用以下配额来满足集群的默认最低要求：

表 9.19. 带有 Kuryr 的 RHOSP 上默认 OpenShift Container Platform 集群的建议资源

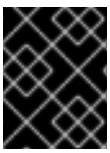
资源	值
浮动 IP 地址	3 - 加上预期的 LoadBalancer 类型服务的数量
端口	1500 - 每个 Pod 需要 1 个
路由器	1
子网	250 - 每个命名空间/项目需要 1 个
网络	250 - 每个命名空间/项目需要 1 个
RAM	112 GB
vCPUs	28
卷存储	275 GB
实例	7
安全组	250 - 每个服务和每个 NetworkPolicy 需要 1 个
安全组规则	1000
负载均衡器	100 - 每个服务需要 1 个
负载均衡器侦听程序	500 - 每个服务公开端口需要 1 个
负载均衡器池	500 - 每个服务公开端口需要 1 个

集群或许能使用少于推荐数量的资源来运作，但其性能无法保证。



重要

如果 RHOSP 对象存储（Swift）可用，并由具有 **swiftoperator** 角色的用户帐户执行，它会作为 OpenShift Container Platform 镜像 registry 的默认后端。在这种情况下，卷存储需要有 175GB。根据镜像 registry 的大小，Swift 空间要求会有所不同。



重要

如果您使用带有 Amphora 驱动而不是 OVN Octavia 驱动的 Red Hat OpenStack Platform (RHOSP) 版本 16，则安全组会与服务帐户而不是用户项目关联。

在设置资源时请考虑以下几点：

- 需要的端口数量会大于 pod 的数量。Kuryr 使用端口池来预创建端口以供 pod 使用，用于加快 pod 的启动时间。

- 每个网络策略都映射到 RHOSP 安全组中，并根据 **NetworkPolicy** 规格将一个或多个规则添加到安全组中。
- 每个服务都映射到一个 RHOSP 负载均衡器中。在估算配额所需安全组数时，请考虑此要求。
  如果您使用 RHOSP 版本 15 或更早版本，或者使用 **ovn-octavia** 驱动，则每个负载均衡器都有一个带有用户项目的安全组。
- 配额不考虑负载均衡器资源（如 VM 资源），但您必须在决定 RHOSP 部署的大小时考虑这些资源。默认安装将有超过 50 个负载均衡器，集群必须可以容纳它们。
  如果您使用启用 OVN Octavia 驱动程序的 RHOSP 版本 16，则只生成一个负载均衡器虚拟机; 服务通过 OVN 流平衡负载。

OpenShift Container Platform 部署由 control plane 机器、计算机器和 bootstrap 机器组成。

要启用 Kuryr SDN，您的环境必须满足以下要求：

- 运行 RHOSP 13+。
- 具有 Octavia 的 Overcloud。
- 使用 Neutron Trunk 端口扩展。
- 如果使用 ML2/OVS Neutron 驱动而不是 **ovs-hybrid**，则请使用 **openvswitch** 防火墙驱动。

9.4.3.1. 增加配额

使用 Kuryr SDN 时，您必须提高配额以满足 pod 、Services 、namespaces 和网络策略所使用的 Red Hat OpenStack Platform（RHOSP）资源要求。

流程

- 运行以下命令为项目增加配额：

```
$ sudo openstack quota set --secgroups 250 --secgroup-rules 1000 --ports 1500 --subnets
250 --networks 250 <project>
```

9.4.3.2. 配置 Neutron

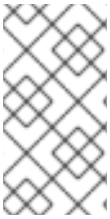
Kuryr CNI 利用 Neutron Trunks 扩展来将容器插入 Red Hat OpenStack Platform（RHOSP）SDN，因此您必须使用 **trunks** 扩展才可以使 Kuryr 正常工作。

另外，如果您使用默认的 ML2/OVS Neutron 驱动程序，防火墙必须设为 **openvswitch** 而不是 **ovs_hybrid** ，以便在中继子端口上强制实施安全组，同时 Kuryr 可以正确处理网络策略。

9.4.3.3. 配置 Octavia

Kuryr SDN 使用 Red Hat OpenStack Platform（RHOSP）的 Octavia LBaaS 来实现 OpenShift Container Platform 服务。因此，您必须在 RHOSP 上安装和配置 Octavia 组件以使用 Kuryr SDN。

要启用 Octavia，您必须在安装 RHOSP Overcloud 的过程中包括 Octavia 服务，如果 Overcloud 已存在则需要升级 Octavia 服务。以下启用 Octavia 的步骤适用于新的 Overcloud 安装或 Overcloud 更新。



注意

以下步骤只包括在部署 RHOSP 时需要处理 Octavia 部分的信息。请注意 registry 可能会不同。

这个示例使用本地的 registry。

流程

1. 如果您使用本地 registry，请创建一个模板来将镜像上传到 registry。例如：

```
(undercloud) $ openstack overcloud container image prepare \
-e /usr/share/openstack-tripleo-heat-templates/environments/services-docker/octavia.yaml \
--namespace=registry.access.redhat.com/rhosp13 \
--push-destination=<local-ip-from-undercloud.conf>:8787 \
--prefix=openstack- \
--tag-from-label {version}-{release} \
--output-env-file=/home/stack/templates/overcloud_images.yaml \
--output-images-file /home/stack/local_registry_images.yaml
```

2. 验证 **local_registry_images.yaml** 文件是否包含 Octavia 镜像。例如：

```
...
- imagename: registry.access.redhat.com/rhosp13/openstack-octavia-api:13.0-43
  push_destination: <local-ip-from-undercloud.conf>:8787
- imagename: registry.access.redhat.com/rhosp13/openstack-octavia-health-manager:13.0-
45
  push_destination: <local-ip-from-undercloud.conf>:8787
- imagename: registry.access.redhat.com/rhosp13/openstack-octavia-housekeeping:13.0-45
  push_destination: <local-ip-from-undercloud.conf>:8787
- imagename: registry.access.redhat.com/rhosp13/openstack-octavia-worker:13.0-44
  push_destination: <local-ip-from-undercloud.conf>:8787
```

注意

Octavia 容器版本根据所安装的特定 RHOSP 版本的不同而有所不同。

3. 将 **registry.redhat.io** 中的容器镜像拉取到 Undercloud 节点：

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

这可能需要一些时间，具体要看您的网络速度和 Undercloud 使用的磁盘。

4. 由于 Octavia 负载均衡器是用来访问 OpenShift Container Platform API，所以您必须增加它们的监听程序的默认超时时间。默认超时为 50 秒。通过将以下文件传递给 Overcloud deploy 命令，将超时时间增加到 20 分钟：

```
(undercloud) $ cat octavia_timeouts.yaml
parameter_defaults:
  OctaviaTimeoutClientData: 1200000
  OctaviaTimeoutMemberData: 1200000
```

注意

RHOSP 13.0.13+ 不需要这一步。

5. 使用 Octavia 安装或更新 overcloud 环境：

```
$ openstack overcloud deploy --templates \
  -e /usr/share/openstack-tripleo-heat-templates/environments/services-docker/octavia.yaml \
  -e octavia_timeouts.yaml
```

注意

这个命令只包含与 Octavia 相关的文件，它根据您具体的 RHOSP 安装而有所不同。如需更多信息，请参阅 RHOSP 文档。有关自定义 Octavia 安装的详情请参考使用 Director 安装 Octavia。

注意

当利用 Kuryr SDN 时，Overcloud 安装需要 Neutron **trunk** 扩展。这在 director 部署中默认可用。当 Neutron 后端是 ML2/OVS 时，使用 **openvswitch** 防火墙而不是默认的 **ovs-hybrid**。如果后端为 ML2/OVN，则不需要修改。

6. 在早于 13.0.13 的 RHOSP 版本中，在创建项目后将项目 ID 添加到 **octavia.conf** 配置文件中。

- 要跨服务实施网络策略，比如网络流量会通过 Octavia 负载均衡器时，您必须确保 Octavia 在用户项目中创建 Amphora VM 安全组。
  这可确保所需的 LoadBalancer 安全组属于该项目，并可将其更新为强制实施服务隔离。

注意

在 RHOSP 13.0.13 或更高版本中不需要此操作。

Octavia 实施新的 ACL API，限制对负载均衡器 VIP 的访问。

a. 获取项目 ID

```
$ openstack project show <project>
```

输出示例

```
+-------------+----------------------------------+
| Field       | Value                            |
+-------------+----------------------------------+
| description |                                  |
| domain_id   | default                          |
| enabled     | True                             |
| id          | PROJECT_ID                       |
| is_domain   | False                            |
| name        | *<project>*                      |
| parent_id   | default                          |
| tags        | []                               |
+-------------+----------------------------------+
```

b. 将项目 ID 添加到控制器的 **octavia.conf** 中。

i. Source **stackrc** 文件：

```
$ source stackrc  # Undercloud credentials
```

ii. 列出 Overcloud 控制器。

```
$ openstack server list
```

输出示例

```
+--------------------------------------+-------------+--------+----------------------+-------
---------+------------+
|
| ID                                   | Name        | Status | Networks
| Image         | Flavor     |
|
+--------------------------------------+-------------+--------+----------------------+-------
---------+------------+
|
| 6bef8e73-2ba5-4860-a0b1-3937f8ca7e01 | controller-0 | ACTIVE |
ctlplane=192.168.24.8 | overcloud-full | controller |
|
| dda3173a-ab26-47f8-a2dc-8473b4a67ab9 | compute-0    | ACTIVE |
ctlplane=192.168.24.6 | overcloud-full | compute    |
|
+--------------------------------------+-------------+--------+----------------------+-------
---------+------------+
```

iii. SSH 到控制器。

```
$ ssh heat-admin@192.168.24.8
```

iv. 编辑 **octavia.conf** 文件，将项目添加到 Amphora 安全组存在于用户账户的项目列表中。

```
# List of project IDs that are allowed to have Load balancer security groups
# belonging to them.
amp_secgroup_allowed_projects = PROJECT_ID
```

c. 重启 Octavia worker 以便重新加载配置。

```
controller-0$ sudo docker restart octavia_worker
```

注意

根据您的 RHOSP 环境，Octavia 可能不支持 UDP 侦听程序。如果您在 RHOSP 版本 13.0.13 或更早版本使用 Kuryr SDN，则不支持 UDP 服务。RHOSP 版本 16 或更高版本支持 UDP。

9.4.3.3.1. Octavia OVN 驱动程序

Octavia 通过 Octavia API 支持多个供应商驱动程序。

要查看所有可用的 Octavia 提供程序驱动，请在命令行中输入：

```
$ openstack loadbalancer provider list
```

输出示例

```
+---------+-------------------------------------------------+
| name    | description                                     |
+---------+-------------------------------------------------+
| amphora | The Octavia Amphora driver.                     |
| octavia | Deprecated alias of the Octavia Amphora driver. |
| ovn     | Octavia OVN driver.                             |
+---------+-------------------------------------------------+
```

从 RHOSP 版本 16 开始，Octavia OVN 供应商驱动程序（**ovn**）在 RHOSP 部署的 OpenShift Container Platform 上被支持。

ovn 是 Octavia 和 OVN 提供的负载平衡集成驱动。它支持基本负载平衡功能，并基于 OpenFlow 规则。在使用 OVN Neutron ML2 的部署中，Director 会在 Octavia 中自动启用该驱动程序。

Amphora 供应商驱动程序是默认驱动程序。如果启用了 **ovn**，Kuryr 将使用它。

如果 Kuryr 使用 **ovn** 而不是 Amphora，则可提供以下优点：

- 资源要求更低Kuryr 不需要为每个服务都提供一个负载均衡器虚拟机。
- 网络延迟会降低。
- 通过对每个服务使用 OpenFlow 规则而不是 VM 来提高服务创建速度。
- 跨所有节点的分布式负载平衡操作，而不是集中到 Amphora 虚拟机中。

9.4.3.4. 已知使用 Kuryr 安装的限制

将 OpenShift Container Platform 与 Kuryr SDN 搭配使用有一些已知的限制。

RHOSP 常规限制
带有 Kuryr SDN 的 OpenShift Container Platform 不支持带有类型 **NodePort** 的 **Service** 对象。

如果机器子网没有连接到路由器，或者子网已连接，但路由器没有设置外部网关，Kuryr 无法为类型为 **LoadBalancer** 的 **Service** 对象创建浮动 IP。

- 在 **Service** 对象上配置 **sessionAffinity=ClientIP** 属性无效。Kuryr 不支持此设置。

RHOSP 版本限制
使用带有 Kuryr SDN 的 OpenShift Container Platform 有一些限制，具体取决于 RHOSP 版本。

- RHOSP 16 之前的版本使用默认 Octavia 负载均衡器驱动程序(Amphora)。此驱动要求在每个 OpenShift Container Platform 服务中部署一个 Amphora 负载均衡器虚拟机。创建太多的服务会导致您耗尽资源。
  如果以后版本的 RHOSP 部署中禁用了 OVN Octavia 驱动程序，则也会使用 Amphora 驱动。它们对资源的要求和早期版本 RHOSP 相同。

- Octavia RHOSP 13.0.13 之前的版本不支持 UDP 侦听程序。因此，OpenShift Container Platform UDP 服务不被支持。
- Octavia RHOSP 13.0.13 之前的版本无法侦听同一端口上的多个协议。不支持将同一端口暴露给不同协议的服务，比如 TCP 和 UDP。
- Kuryr SDN 不支持由服务自动取消闲置。

RHOSP 环境限制
使用取决于您的部署环境的 Kuryr SDN 会有一些限制。

由于 Octavia 缺少对 UDP 协议和多个监听器的支持，如果 rhosp 版本早于 13.0.13，Kuryr 会强制 pod 在 DNS 解析中使用 TCP，如果：

在 Go 版本 1.12 及更早的版本中，通过 CGO 支持被禁用的模式编译的应用程序只使用 UDP。在这种情况下，native Go 解析器无法识别 **resolv.conf** 中的 **use-vc** 选项，它控制 DNS 解析是否强制使用 TCP。因此，UDP 仍会被用来解析 DNS，这将导致失败。

要确保 TCP 强制使用是允许的，在编译应用程序使把环境变量 **CGO_ENABLED** 设定为 **1**（如 **CGO_ENABLED=1**），或者不使用这个变量。

在 Go 版本 1.13 及之后的版本中，如果使用 UDP 的 DNS 解析失败，则会自动使用 TCP。

注意

基于 musl 的容器，包括基于 Alpine 的容器，不支持 **use-vc** 选项。

RHOSP 升级限制
作为 RHOSP 升级过程的结果，可能会更改 Octavia API，并可能需要升级到用于负载均衡器的 Amphora 镜像。

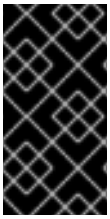
您可以单独处理 API 更改。

如果升级了 Amphora 镜像，RHOSP Operator 可使用两种方式处理现有的负载均衡器虚拟机：

- 通过触发负载均衡器故障切换来升级每个虚拟机。
- 将升级虚拟机的职责留给用户。

如果运算符使用第一个选项，在故障切换过程中可能会有短暂的停机时间。

如果 Operator 采用第二个选项，现有负载均衡器将不支持升级的 Octavia API 功能，比如 UDP 侦听程序。在这种情况下，用户必须重新创建自己的服务以使用这些功能。



重要

如果 OpenShift Container Platform 检测到支持 UDP 负载均衡的新 Octavia 版本，它会自动重新创建 DNS 服务。服务重新创建可确保服务默认支持 UDP 负载均衡。

这个重新创建会导致 DNS 服务大约停机 一 分钟。

9.4.3.5. control plane 机器

默认情况下，OpenShift Container Platform 安装过程会创建三台 control plane 机器。

每台机器都需要：

- 来自 RHOSP 配额的实例
- 来自 RHOSP 配额的端口
- 至少 16 GB 内存、4 个 vCPU 和 100 GB 存储空间的类别

9.4.3.6. 计算机器

默认情况下，OpenShift Container Platform 安装过程会创建三台计算机器。

每台机器都需要：

- 来自 RHOSP 配额的实例
- 来自 RHOSP 配额的端口
- 至少有 8 GB 内存、2 个 vCPU 和 100 GB 存储空间的类别

提示

计算机器托管您在 OpenShift Container Platform 上运行的应用程序；运行数量应尽可能多。

9.4.3.7. bootstrap 机器

在安装时，会临时置备 bootstrap 机器来支持 control plane。生产控制平面就绪后，bootstrap 机器会被取消置备。

bootstrap 机器需要：

- 来自 RHOSP 配额的实例
- 来自 RHOSP 配额的端口
- 至少 16 GB 内存、4 个 vCPU 和 100 GB 存储空间的类别

9.4.4. OpenShift Container Platform 的互联网访问

在 OpenShift Container Platform 4.6 中，您需要访问互联网来安装集群。

您必须具有以下互联网访问权限：

- 访问 OpenShift Cluster Manager 以下载安装程序并执行订阅管理。如果集群可以访问互联网，并且没有禁用 Telemetry，该服务会自动授权您的集群。
- 访问 Quay.io，以获取安装集群所需的软件包。
- 获取执行集群更新所需的软件包。

重要

如果您的集群无法直接访问互联网，则可以在置备的某些类基础架构上执行受限网络安装。在此过程中，您要下载所需的内容，并使用它在镜像 registry（mirror registry） 中填充安装集群并生成安装程序所需的软件包。对于某些安装类型，集群要安装到的环境不需要访问互联网。在更新集群之前，要更新 registry 镜像系统中的内容。

9.4.5. 下载 playbook 的依赖项

简化用户置备基础架构安装过程的 Ansible playbook 需要几个 Python 模块。在您要运行安装程序的机器上添加模块的仓库，然后下载它们。

注意

这些说明假设您使用 Red Hat Enterprise Linux（RHEL）8。

先决条件

- Python 3 已安装在您的机器上。

流程

1. 在命令行中添加软件仓库：
   a. 使用 Red Hat Subscription Manager 注册：

```
$ sudo subscription-manager register # If not done already
```

   b. 获取最新的订阅数据：

```
$ sudo subscription-manager attach --pool=$YOUR_POOLID # If not done already
```

   c. 禁用当前的软件仓库：

```
$ sudo subscription-manager repos --disable=* # If not done already
```

   d. 添加所需的软件仓库：

```
$ sudo subscription-manager repos \
  --enable=rhel-8-for-x86_64-baseos-rpms \
  --enable=openstack-16-tools-for-rhel-8-x86_64-rpms \
  --enable=ansible-2.9-for-rhel-8-x86_64-rpms \
  --enable=rhel-8-for-x86_64-appstream-rpms
```

2. 安装模块：

```
$ sudo yum install python3-openstackclient ansible python3-openstacksdk python3-netaddr
```

3. 确保 **python** 命令指向 **python3**:

```
$ sudo alternatives --set python /usr/bin/python3
```

9.4.6. 下载安装 playbook

下载 Ansible playbook，可用于在您自己的 Red Hat OpenStack Platform（RHOSP）基础架构上安装 OpenShift Container Platform。

先决条件

- curl 命令行工具可在您的机器上找到。

流程

- 要将 playbook 下载到您的工作目录中，请从命令行运行以下脚本：

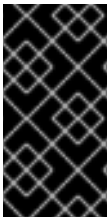
```
$ xargs -n 1 curl -O <<< '
    https://raw.githubusercontent.com/openshift/installer/release-4.6/upi/openstack/bootstrap.yaml
    https://raw.githubusercontent.com/openshift/installer/release-4.6/upi/openstack/common.yaml
    https://raw.githubusercontent.com/openshift/installer/release-4.6/upi/openstack/compute-nodes.yaml
    https://raw.githubusercontent.com/openshift/installer/release-4.6/upi/openstack/control-plane.yaml
    https://raw.githubusercontent.com/openshift/installer/release-4.6/upi/openstack/inventory.yaml
    https://raw.githubusercontent.com/openshift/installer/release-4.6/upi/openstack/network.yaml
    https://raw.githubusercontent.com/openshift/installer/release-4.6/upi/openstack/security-groups.yaml
    https://raw.githubusercontent.com/openshift/installer/release-4.6/upi/openstack/down-bootstrap.yaml
    https://raw.githubusercontent.com/openshift/installer/release-4.6/upi/openstack/down-compute-nodes.yaml
    https://raw.githubusercontent.com/openshift/installer/release-4.6/upi/openstack/down-control-plane.yaml
    https://raw.githubusercontent.com/openshift/installer/release-4.6/upi/openstack/down-load-balancers.yaml
    https://raw.githubusercontent.com/openshift/installer/release-4.6/upi/openstack/down-network.yaml
    https://raw.githubusercontent.com/openshift/installer/release-4.6/upi/openstack/down-security-groups.yaml
    https://raw.githubusercontent.com/openshift/installer/release-4.6/upi/openstack/down-containers.yaml'
```

playbook 下载到您的机器中。

重要

在安装过程中，您可以修改 playbook 来配置部署。

在集群生命周期中保留所有 playbook。您必须具有 playbook，才能从 RHOSP 中删除 OpenShift Container Platform 集群。



重要

您在 **bootstrap.yaml**、**compute-nodes.yaml**、**control-plane.yaml**、**network.yaml** 和 **security-groups.yaml** 文件中进行的任何改变都需要与带有 **down-** 前缀的对应的 playbook 相匹配。例如，对 **bootstrap.yaml** 文件的编辑也必须反映在 **down-bootstrap.yaml** 文件中。如果没有编辑这两个文件，则支持的删除集群过程将失败。

9.4.7. 获取安装程序

在安装 OpenShift Container Platform 之前，将安装文件下载到本地计算机上。

先决条件

- 运行 Linux 或 macOS 的计算机，本地磁盘空间为 500 MB

流程

1. 访问 OpenShift Cluster Manager 站点的 Infrastructure Provider 页面。如果您有红帽帐号，请使用自己的凭证登录。如果没有，请创建一个帐户。
2. 选择您的基础架构供应商。
3. 进入适用于您的安装类型的页面，下载您的操作系统的安装程序，并将文件放在要保存安装配置文件的目录中。。

重要

安装程序会在用来安装集群的计算机上创建若干文件。在完成集群安装后，您必须保留安装程序和安装程序所创建的文件。这两个文件都需要删除集群。

重要

删除安装程序创建的文件不会删除您的集群，即使集群在安装过程中失败也是如此。要删除集群，为特定云供应商完成 OpenShift Container Platform 卸载流程。

4. 提取安装程序。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ tar xvf openshift-install-linux.tar.gz
```

5. 从 Red Hat OpenShift Cluster Manager 下载安装 pull secret 。通过此 pull secret，您可以进行所含授权机构提供的服务的身份验证，这些服务包括为 OpenShift Container Platform 组件提供容器镜像的 Quay.io。

9.4.8. 生成 SSH 私钥并将其添加到代理中

如果要在集群上执行安装调试或灾难恢复，则必须为 **ssh-agent** 和安装程序提供 SSH 密钥。您可以使用此密钥访问公共集群中的 bootstrap 机器来排除安装问题。

注意

在生产环境中，您需要进行灾难恢复和调试。

您可以使用此密钥以 **core** 用户身份通过 SSH 连接到 master 节点。在部署集群时，此密钥会添加到 **core** 用户的 **~/.ssh/authorized_keys** 列表中。

流程

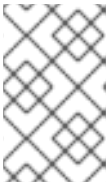
1. 如果还没有为计算机上免密码身份验证而配置的 SSH 密钥，请创建一个。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ ssh-keygen -t ed25519 -N '' \
  -f <path>/<file_name> 1
```

-

1 指定新 SSH 密钥的路径和文件名，如 ~/**.ssh/id_rsa**。如果您已有密钥对，请确保您的公钥位于 ~/**.ssh** 目录中。

运行此命令会在指定的位置生成不需要密码的 SSH 密钥。



注意

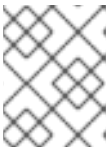
如果您计划在 **x86_64** 架构中安装使用 FIPS 验证的/Modules in Process 加密库的 OpenShift Container Platform 集群，不要创建使用 **ed25519** 算法的密钥。反之，创建一个使用 **rsa** 或 **ecdsa** 算法的密钥。

2. 作为后台任务启动 **ssh-agent** 进程：

```
$ eval "$(ssh-agent -s)"
```

输出示例

```
Agent pid 31874
```



注意

如果您的集群采用 FIPS 模式，则只使用 FIPS 兼容算法来生成 SSH 密钥。密钥必须是 RSA 或 ECDSA。

3. 将 SSH 私钥添加到 **ssh-agent**：

```
$ ssh-add <path>/<file_name> 1
```

输出示例

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

1 指定 SSH 私钥的路径和文件名，如 ~/**.ssh/id_rsa**

后续步骤

- 在安装 OpenShift Container Platform 时，为安装程序提供 SSH 公钥。

9.4.9. 创建 Red Hat Enterprise Linux CoreOS (RHCOS) 镜像

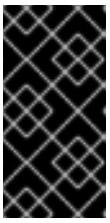
OpenShift Container Platform 安装程序要求 Red Hat OpenStack Platform (RHOSP) 集群中有 Red Hat Enterprise Linux CoreOS (RHCOS) 镜像。检索最新的 RHCOS 镜像，然后使用 RHOSP CLI 上传该镜像。

先决条件

- 已安装了 RHOSP CLI。

流程

1. 登录到红帽客户门户网站的产品下载页。
2. 在 **Version** 下，为 Red Hat Enterprise Linux（RHEL）8 选择 OpenShift Container Platform 4.6 的最新发行版本。



重要

RHCOS 镜像可能不会随着 OpenShift Container Platform 的每一发行版本都有改变。您必须下载最高版本的镜像，其版本号应小于或等于您安装的 OpenShift Container Platform 版本。如果可用，请使用与 OpenShift Container Platform 版本匹配的镜像版本。

3. 下载 *Red Hat Enterprise Linux CoreOS（RHCOS） - OpenStack Image (QCOW)* 。
4. 解压镜像。

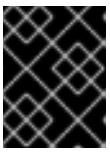
注意

您必须解压 RHOSP 镜像，然后集群才能使用它。下载的文件名可能不包含压缩扩展名，如 **.gz** 或 **.tgz**。要找出是否或者如何压缩文件，请在命令行中输入：

```
$ file <name_of_downloaded_file>
```

5. 从您下载的镜像，使用 RHOSP CLI 在集群中创建名为 **rhcos** 的镜像：

```
$ openstack image create --container-format=bare --disk-format=qcow2 --file rhcos-${RHCOS_VERSION}-openstack.qcow2 rhcos
```



重要

根据您的 RHOSP 环境，可能需要使用 **.raw** 或 **.qcow2** 格式下载镜像。如果使用 Ceph，则必须使用 **.raw** 格式。

警告

如果安装程序发现多个同名的镜像，它会随机选择其中之一。为避免这种行为，请在 RHOSP 中为资源创建唯一名称。

将镜像上传到 RHOSP 后，就可以被安装程序使用。

9.4.10. 验证外部网络访问

OpenShift Container Platform 安装进程需要外部网络访问权限。您必须为其提供外部网络值，否则部署会失败。在运行安装进程前，请验证 Red Hat OpenStack Platform (RHOSP) 中是否存在具有外部路由器类型的网络。

先决条件

- 将 OpenStack 联网服务配置为使用 DHCP 代理转发实例 DNS 查询

流程

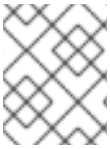
1. 使用 RHOSP CLI 验证"外部"网络的名称和 ID：

```
$ openstack network list --long -c ID -c Name -c "Router Type"
```

输出示例

```
+--------------------------------------+---------------+-------------+
| ID                                   | Name          | Router Type |
+--------------------------------------+---------------+-------------+
| 148a8023-62a7-4672-b018-003462f8d7dc | public_network | External    |
+--------------------------------------+---------------+-------------+
```

网络列表中会显示具有外部路由器类型的网络。如果最少有一个没有，请参阅 创建默认浮动 IP 网络和创建默认供应商网络。



注意

如果启用了 Neutron 中继服务插件，则默认创建中继端口。如需更多信息，请参阅 Neutron 中继端口。

9.4.11. 启用对环境的访问

在部署时，所有 OpenShift Container Platform 机器都是在 Red Hat OpenStack Platform (RHOSP) 租户网络中创建的。因此，大多数 RHOSP 部署中都无法直接访问它们。

您可以在安装过程中使用浮动 IP 地址（FIP）来配置 OpenShift Container Platform API 和应用程序访问。您也可以在没有配置 FIP 的情况下完成安装，但安装程序不会配置一种从外部访问 API 或应用程序的方法。

9.4.11.1. 启用通过浮动 IP 地址进行访问

创建浮动 IP(FIP)地址，用于从外部访问 OpenShift Container Platform API、集群应用程序和 bootstrap 过程。

流程

1. 使用 Red Hat OpenStack Platform (RHOSP) CLI，创建 API FIP：

```
$ openstack floating ip create --description "API <cluster_name>.<base_domain>"
<external_network>
```

2. 使用 Red Hat OpenStack Platform (RHOSP) CLI，创建应用程序或 Ingress，FIP：

```
$ openstack floating ip create --description "Ingress <cluster_name>.<base_domain>"
<external_network>
```

3. 使用 Red Hat OpenStack Platform（RHOSP）CLI 创建 bootstrap FIP:

```
$ openstack floating ip create --description "bootstrap machine" <external_network>
```

4. 向用于 API 和 Ingress FIP 的 DNS 服务器添加符合这些模式的记录：

```
api.<cluster_name>.<base_domain>.  IN  A  <API_FIP>
*.apps.<cluster_name>.<base_domain>. IN  A <apps_FIP>
```

注意

如果您不控制 DNS 服务器，您可以通过将集群域名（如以下内容）添加到 **/etc/hosts** 文件中来访问集群：

- **<api_floating_ip> api.<cluster_name>.<base_domain>**
- **<application_floating_ip> grafana-openshift-monitoring.apps.<cluster_name>.<base_domain>**
- **<application_floating_ip> prometheus-k8s-openshift-monitoring.apps.<cluster_name>.<base_domain>**
- **<application_floating_ip> oauth-openshift.apps.<cluster_name>.<base_domain>**
- **<application_floating_ip> console-openshift-console.apps.<cluster_name>.<base_domain>**
- **application_floating_ip integrate-oauth-server-openshift-authentication.apps.<cluster_name>.<base_domain>**

/etc/hosts 文件中的集群域名授予对本地集群的 Web 控制台和监控界面的访问权限。您还可以使用 **kubectl** 或 **oc**。您可以使用指向 <application_floating_ip> 的额外条目来访问用户应用程序。此操作使 API 和应用程序可供您访问，不适合于生产部署，但允许对开发和测试进行安装。

5. 将 FIP 添加到 **inventory.yaml** 文件，作为以下变量的值：
   - **os_api_fip**
   - **os_bootstrap_fip**
   - **os_ingress_fip**

如果使用这些值，还必须在 **inventory.yaml** 文件中输入一个外部网络作为 **os_external_network** 变量的值。

提示

您可以通过分配浮动 IP 地址并更新防火墙配置，使 OpenShift Container Platform 资源在集群之外可用。

9.4.11.2. 完成没有浮动 IP 地址的安装

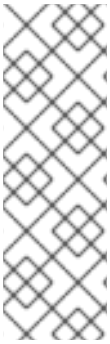
您可以在不提供浮动 IP 地址的情况下在 Red Hat OpenStack Platform（RHOSP）上安装 OpenShift Container Platform。

在 **inventory.yaml** 文件中，不要定义以下变量：

- **os_api_fip**
- **os_bootstrap_fip**
- **os_ingress_fip**

如果无法提供外部网络，也可以将 **os_external_network** 留空。如果没有为 **os_external_network** 提供值，则不会为您创建路由器。如果没有额外的操作，安装程序将无法从 Glance 检索镜像。之后在安装过程中，当您创建网络资源时，必须自行配置外部连接。

如果您使用 **wait-for** 命令从因为缺少浮动 IP 地址或名称解析而无法访问集群 API 的系统中运行安装程序时，安装会失败。要防止安装失败，可以使用代理网络或者从与您的机器位于同一网络的系统中运行安装程序。



注意

您可以通过为 API 和 Ingress 端口创建 DNS 记录来启用名称解析。例如：

```
api.<cluster_name>.<base_domain>.  IN  A  <api_port_IP>
*.apps.<cluster_name>.<base_domain>. IN  A <ingress_port_IP>
```

如果您不控制 DNS 服务器，可以改为将记录添加到 **/etc/hosts** 文件中。此操作使 API 可供您自己访问，不适合于生产部署。这可用于进行开发和测试的安装。

9.4.12. 为安装程序定义参数

OpenShift Container Platform 安装程序依赖于一个名为 **clouds.yaml** 的文件。该文件描述了 Red Hat OpenStack Platform (RHOSP) 配置参数，包括项目名称、登录信息和授权服务 URL。

流程

1. 创建 **clouds.yaml** 文件：
   - 如果您的 RHOSP 发行版包含 Horizon web UI，请在该 UI 中生成 **clouds.yaml** 文件。

     重要

     请记住在 **auth** 字段中添加密码。您也可以把 secret 保存在 **clouds.yaml** 以外的一个独立的文件中。

   - 如果您的 RHOSP 发行版不包含 Horizon Web UI，或者您不想使用 Horizon，请自行创建该文件。如需有关 **clouds.yaml** 的详细信息，请参阅 RHOSP 文档中的配置文件。

     ```
     clouds:
       shiftstack:
         auth:
           auth_url: http://10.10.14.42:5000/v3
           project_name: shiftstack
           username: shiftstack_user
           password: XXX
           user_domain_name: Default
           project_domain_name: Default
     ```

```
dev-env:
  region_name: RegionOne
  auth:
    username: 'devuser'
    password: XXX
    project_name: 'devonly'
    auth_url: 'https://10.10.14.22:5001/v2.0'
```

2. 如果您的 RHOSP 安装使用自签名证书颁发机构 (CA) 证书进行端点身份验证：

   a. 将 CA 文件复制到您的机器中。

   b. 将机器添加到证书颁发机构信任捆绑包中：

   ```
   $ sudo cp ca.crt.pem /etc/pki/ca-trust/source/anchors/
   ```

   c. 更新信任捆绑包：

   ```
   $ sudo update-ca-trust extract
   ```

   d. 将 **cacerts** 键添加到 **clouds.yaml** 文件。该值必须是到 CA 证书的绝对路径，则其可以被非根用户访问：

   ```
   clouds:
     shiftstack:
       ...
       cacert: "/etc/pki/ca-trust/source/anchors/ca.crt.pem"
   ```

   提示

   使用自定义 CA 证书运行安装程序后，您可以通过编辑 **cloud-provider-config** keymap 中的**ca-cert.pem** 键的值来更新证书。在命令行中运行：

   ```
   $ oc edit configmap -n openshift-config cloud-provider-config
   ```

3. 将 **clouds.yaml** 文件放在以下位置之一：

   a. **OS_CLIENT_CONFIG_FILE** 环境变量的值

   b. 当前目录

   c. 特定于 Unix 的用户配置目录，如 **~/.config/openstack/clouds.yaml**

   d. 特定于 Unix 的站点配置目录，如 **/etc/openstack/clouds.yaml**
   安装程序会按照以上顺序搜索 **clouds.yaml**。

9.4.13. 创建安装配置文件

您可以自定义在 Red Hat OpenStack Platform（RHOSP）上安装的 OpenShift Container Platform 集群。

先决条件

- 获取 OpenShift Container Platform 安装程序以及集群的 pull secret。

流程

1. 创建 **install-config.yaml** 文件。
   a. 更改到包含安装程序的目录，再运行以下命令：

```
$ ./openshift-install create install-config --dir <installation_directory> 1
```

   1 对于 **<installation_directory>**，请指定用于保存安装程序所创建的文件的目录名称。

重要

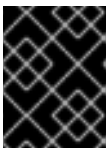
指定一个空目录。一些安装信息，如 bootstrap X.509 证书，有较短的过期间隔，因此不要重复使用安装目录。如果要重复使用另一个集群安装中的个别文件，可以将其复制到您的目录中。但是，一些安装数据的文件名可能会在发行版本之间有所改变。从 OpenShift Container Platform 老版本中复制安装文件时要格外小心。

   b. 在提示符处，提供您的云的配置详情：
      i. 可选：选择用来访问集群机器的 SSH 密钥。

注意

对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 **ssh-agent** 进程使用的 SSH 密钥。

      ii. 选择 **openstack** 作为目标平台。
      iii. 指定用于安装集群的 Red Hat OpenStack Platform (RHOSP) 外部网络名称。
      iv. 指定用于从外部访问 OpenShift API 的浮动 IP 地址。
      v. 指定至少有 16 GB RAM 用于 control plane 节点，以及计算节点的 8 GB RAM。
      vi. 选择集群要部署到的基域。所有 DNS 记录都将是这个基域的子域，并包含集群名称。
      vii. 为集群输入一个名称。名称不能多于 14 个字符。
      viii. 粘贴 Red Hat OpenShift Cluster Manager 中的 pull secret 。
2. 修改 **install-config.yaml** 文件。您可以在**安装配置参数**部分中找到有关可用参数的更多信息。
3. 备份 **install-config.yaml** 文件，以便用于安装多个集群。



重要

install-config.yaml 文件会在安装过程中消耗掉。如果要重复使用此文件，必须现在备份。

现在，文件 **install-config.yaml** 位于您指定的目录中。

9.4.14. 安装配置参数

在部署 OpenShift Container Platform 集群前，您可以提供参数值，以描述托管集群的云平台的帐户并选择性地自定义集群平台。在创建 **install-config.yaml** 安装配置文件时，您可以通过命令行来提供所需的参数的值。如果要自定义集群，可以修改 **install-config.yaml** 文件来提供关于平台的更多信息。

注意

安装之后，您无法修改 **install-config.yaml** 文件中的这些参数。

重要

openshift-install 命令不验证参数的字段名称。如果指定了不正确的名称，则不会创建相关的文件或对象，且不会报告错误。确保所有指定的参数的字段名称都正确。

9.4.14.1. 所需的配置参数

下表描述了所需的安装配置参数：

表 9.20. 所需的参数

参数	描述	值
apiVersion	**install-config.yaml** 内容的 API 版本。当前版本是 **v1**。安装程序还可能支持旧的 API 版本。	字符串
baseDomain	云供应商的基域。此基础域用于创建到 OpenShift Container Platform 集群组件的路由。集群的完整 DNS 名称是 **baseDomain** 和 **metadata.name** 参数值的组合，其格式为 **<metadata.name>.<baseDomain>**。	完全限定域名或子域名，如 **example.com**。
metadata	Kubernetes 资源 **ObjectMeta**，其中只消耗 **name** 参数。	对象
metadata.name	集群的名称。集群的 DNS 记录是 **{{.metadata.name}}.{{.baseDomain}}** 的子域。	小写字母,连字符(-)和句点(.)的字符串，如 **dev**。该字符串长度必须为 14 个字符或更少。
platform	执行安装的具体平台配置：**aws**、**baremetal**、**azure**、**openstack**、**ovirt**、**vsphere**。有关 **platform.<platform>** 参数的额外信息，请参考下表来了解您的具体平台。	对象

参数	描述	值

pullSecret	从 Red Hat OpenShift Cluster Manager 获取 pull secret，验证从 Quay.io 等服务中下载 OpenShift Container Platform 组件的容器镜像。	<pre>{
		  "auths":{
		  "cloud.openshift.com":{
		  "auth":"b3Blb=",
		  "email":"you@example.com"
		  },
		  "quay.io":{
		  "auth":"b3Blb=",
		  "email":"you@example.com"
		  }
		  }
		}</pre>

9.4.14.2. 网络配置参数

您可以根据现有网络基础架构的要求自定义安装配置。例如，您可以扩展集群网络的 IP 地址块，或者提供不同于默认值的不同 IP 地址块。

只支持 IPv4 地址。

表 9.21. 网络参数

参数	描述	值
networking	集群网络的配置。	对象 **注意** 您不能在安装后修改 **networking** 对象指定的参数。
networking.networkType	要安装的集群网络供应商 Container Network Interface（CNI）插件。	**OpenShiftSDN** 或 **OVNKubernetes**。默认值为 **OpenShiftSDN**。

networking.clusterNetwork	pod 的 IP 地址块。
	
	默认值为 **10.128.0.0/14**，主机前缀为 **/23**。
	
	如果您指定多个 IP 地址块，则块不得互相重叠。
		一个对象数组。例如：
		
		<pre>networking:
		  clusterNetwork:
		  - cidr: 10.128.0.0/14
		  hostPrefix: 23</pre>

networking.clusterNetwork.cidr	使用 **networking.clusterNetwork** 时需要此项。IP 地址块。 一个 IPv4 网络。	使用 CIDR 形式的 IP 地址块。IPv4 块的前缀长度介于 **0** 到 **32** 之间。

参数	描述	值
networking.clusterNetwork.hostPrefix	分配给每个单独节点的子网前缀长度。例如，如果 **hostPrefix** 设为 **23**，则每个节点从所给的 **cidr** 中分配一个 **/23** 子网。**hostPrefix** 值 **23** 提供 510（2^(32 - 23)- 2）个 pod IP 地址。	子网前缀。 默认值为 **23**。
networking.serviceNetwork	服务的 IP 地址块。默认值为 **172.30.0.0/16**。 OpenShift SDN 和 OVN-Kubernetes 网络供应商只支持服务网络的一个 IP 地址块。	CIDR 格式具有 IP 地址块的数组。例如： networking: serviceNetwork: - 172.30.0.0/16
networking.machineNetwork	机器的 IP 地址块。 如果您指定多个 IP 地址块，则块不得互相重叠。	一个对象数组。例如： networking: machineNetwork: - cidr: 10.0.0.0/16
networking.machineNetwork.cidr	使用 **networking.machineNetwork** 时需要。IP 地址块。libvirt 以外的所有平台的默认值为 **10.0.0.0/16**。对于 libvirt，默认值为 **192.168.126.0/24**。	CIDR 表示法中的 IP 网络块。 例如： **10.0.0.0/16**。 **注意** 将 **networking.machineNetwork** 设置为与首选 NIC 所在的 CIDR 匹配。

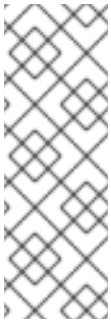
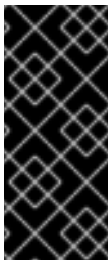
9.4.14.3. 可选配置参数

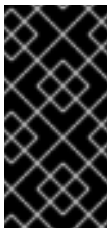
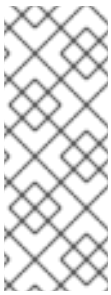
下表描述了可选安装配置参数：

表 9.22. 可选参数

参数	描述	值
additionalTrustBundle	添加到节点可信证书存储中的 PEM 编码 X.509 证书捆绑包。配置了代理时，也可以使用这个信任捆绑包。	字符串
compute	组成计算节点的机器的配置。	machine-pool 对象的数组。详情请查看以下"Machine-pool"表。

参数	描述	值
compute.architecture	决定池中机器的指令集合架构。目前不支持异构集群，因此所有池都必须指定相同的架构。有效值为 **amd64** （默认值）。	字符串
compute.hyperthreading	是否在计算机器上启用或禁用并发多线程或**超线程**。默认情况下，启用并发多线程以提高机器内核的性能。  **重要** 如果禁用并发多线程，请确保在容量规划时考虑到机器性能可能会显著降低的问题。	**Enabled** 或 **Disabled**
compute.name	使用 **compute** 时需要此值。机器池的名称。	**worker**
compute.platform	使用 **compute** 时需要此值。使用此参数指定托管 worker 机器的云供应商。此参数值必须与 **controlPlane.platform** 参数值匹配。	**aws**、**azure**、**gcp**、**openstack**、**ovirt**、**vsphere** 或 **{}**
compute.replicas	要置备的计算机器数量，也称为 worker 机器。	大于或等于 **2** 的正整数。默认值为 **3**。
controlPlane	组成 control plane 的机器的配置。	**MachinePool** 对象的数组。详情请查看以下"Machine-pool"表。
controlPlane.architecture	决定池中机器的指令集合架构。目前不支持异构集群，因此所有池都必须指定相同的架构。有效值为 **amd64** （默认值）。	字符串
controlPlane.hyperthreading	是否在 control plane 机器上启用或禁用并发多线程或**超线程**。默认情况下，启用并发多线程以提高机器内核的性能。  **重要** 如果禁用并发多线程，请确保在容量规划时考虑到机器性能可能会显著降低的问题。	**Enabled** 或 **Disabled**

参数	描述	值
controlPlane.name	使用 **controlPlane** 时需要。机器池的名称。	**master**
controlPlane.platform	使用 **controlPlane** 时需要。使用此参数指定托管 control plane 机器的云供应商。此参数值必须与 **compute.platform** 参数值匹配。	**aws**、**azure**、**gcp**、**openstack**、**ovirt**、**vsphere** 或 **{}**
controlPlane.replicas	要置备的 control plane 机器数量。	唯一支持的值是 **3**，它是默认值。
credentialsMode	Cloud Credential Operator（CCO）模式。如果没有指定任何模式，CCO 会动态地尝试决定提供的凭证的功能，在支持多个模式的平台上使用 mint 模式。  **注意** 不是所有 CCO 模式都支持所有云供应商。如需有关 CCO 模式的更多信息，请参阅*Red Hat Operator 参考指南*内容中的 *Cloud Credential Operator* 条目。	**Mint**、**Passthrough**、**Manual** 或空字符串(**""**)。
fips	启用或禁用 FIPS 模式。默认为 **false**（禁用）。如果启用了 FIPS 模式，运行 OpenShift Container Platform 的 Red Hat Enterprise Linux CoreOS (RHCOS) 机器会绕过默认的 Kubernetes 加密套件，并使用由 RHCOS 提供的加密模块。  **重要** 只有在 **x86_64** 架构中的 OpenShift Container Platform 部署支持 FIPS 验证的/Modules in Process 加密库。  **注意** 如果使用 Azure File 存储，则无法启用 FIPS 模式。	**false** 或 **true**

参数	描述	值
imageContentSources	release-image 内容的源和仓库。	对象数组。包括一个 **source** 以及可选的 **mirrors**，如下表所示。
imageContentSources.source	使用 **imageContentSources** 时需要。指定用户在镜像拉取规格中引用的仓库。	字符串
imageContentSources.mirrors	指定可能还包含同一镜像的一个或多个仓库。	字符串数组
publish	如何发布或公开集群的面向用户的端点，如 Kubernetes API、OpenShift 路由。	**Internal** 或 **External**。默认值为 **External**。 在非云平台上不支持将此字段设置为 **Internal**。  **重要** 如果将字段的值设为 **Internal**，集群将无法运行。如需更多信息，请参阅 BZ#1953035。
sshKey	用于验证集群机器访问的 SSH 密钥或密钥。  **注意** 对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 **ssh-agent** 进程使用的 SSH 密钥。	一个或多个密钥。例如： sshKey: <key1> <key2> <key3>

9.4.14.4. 其他 Red Hat OpenStack Platform（RHOSP）配置参数

下表描述了其他 RHOSP 配置参数：

表 9.23. 其他 RHOSP 参数

参数	描述	值
compute.platform.openstack.rootVolume.size	对于计算机器，以 GB 为单位表示的根卷大小。如果您不设置这个值，机器将使用临时存储。	整数，如 **30**。
compute.platform.openstack.rootVolume.type	对于计算机器，根卷的类型。	字符串，如 **performance**。
controlPlane.platform.openstack.rootVolume.size	对于 control plane 机器，以 GB 为单位表示的根卷大小。如果您不设置这个值，机器将使用临时存储。	整数，如 **30**。
controlPlane.platform.openstack.rootVolume.type	对于 control plane 机器，根卷的类型。	字符串，如 **performance**。
platform.openstack.cloud	要使用的 RHOSP 云的名称，来自于 **clouds.yaml** 文件中的云列表。	字符串，如 **MyCloud**。
platform.openstack.externalNetwork	用于安装的 RHOSP 外部网络名称。	字符串，如 **external**。
platform.openstack.computeFlavor	用于 control plane 和计算机器的 RHOSP 类别。	字符串，如 **m1.xlarge**。

9.4.14.5. 可选 RHOSP 配置参数

下表描述了可选 RHOSP 配置参数：

表 9.24. 可选的 RHOSP 参数

参数	描述	值
compute.platform.openstack.additionalNetworkIDs	与计算机器关联的其他网络。不能为额外网络创建允许的地址对。	一个或多个 UUID 列表作为字符串。例如：**fa806b2f-ac49-4bce-b9db-124bc64209bf**。
compute.platform.openstack.additionalSecurityGroupIDs	与计算机器关联的其他安全组。	一个或多个 UUID 列表作为字符串。例如：**7ee219f3-d2e9-48a1-96c2-e7429f1b0da7**。

参数	描述	值
compute.platform.openstack.zones	RHOSP Compute（Nova）可用区（AZ）在其中安装机器。如果没有设置此参数，安装程序会依赖于配置了 RHOSP 管理员的 Nova 的默认设置。 在使用 Kuryr 的集群上，RHOSP Octavia 不支持可用域。负载均衡器，如果您使用 Amphora 供应商驱动程序，则依赖 Amphora 虚拟机的 OpenShift Container Platform 服务不会根据此属性的值创建。	字符串列表。例如：**["zone-1", "zone-2"]**。
controlPlane.platform.openstack.additionalNetworkIDs	与 control plane 机器关联的额外网络。不能为额外网络创建允许的地址对。	一个或多个 UUID 列表作为字符串。例如：**fa806b2f-ac49-4bce-b9db-124bc64209bf**。
controlPlane.platform.openstack.additionalSecurityGroupIDs	与 control plane 机器关联的其他安全组。	一个或多个 UUID 列表作为字符串。例如：**7ee219f3-d2e9-48a1-96c2-e7429f1b0da7**。
controlPlane.platform.openstack.zones	RHOSP Compute（Nova）可用区（AZ）在其中安装机器。如果没有设置此参数，安装程序会依赖于配置了 RHOSP 管理员的 Nova 的默认设置。 在使用 Kuryr 的集群上，RHOSP Octavia 不支持可用域。负载均衡器，如果您使用 Amphora 供应商驱动程序，则依赖 Amphora 虚拟机的 OpenShift Container Platform 服务不会根据此属性的值创建。	字符串列表。例如：**["zone-1", "zone-2"]**。
platform.openstack.clusterOSImage	安装程序从中下载 RHCOS 镜像的位置。 您必须设置此参数以便在受限网络中执行安装。	HTTP 或 HTTPS URL，可选使用 SHA-256 checksum。 例如：**http://mirror.example.com/images/rhcos-43.81.201912131630.0-openstack.x86_64.qcow2.gz?sha256=ffebbd68e8a1f2a245ca19522c16c86f67f9ac8e4e0c1f0a812b068b16f7265d**。该值也可以是现有 Glance 镜像的名称，如 **my-rhcos**。

参数	描述	值

platform.openstack.defaultMachinePlatform	默认机器池平台配置。	<pre>{
		  "type": "ml.large",
		  "rootVolume": {
		  "size": 30,
		  "type": "performance"
		  }
		}</pre>

platform.openstack.ingressFloatingIP	与 Ingress 端口关联的现有浮动 IP 地址。要使用此属性，还必须定义 **platform.openstack.externalNetwork** 属性。	IP 地址，如 **128.0.0.1**。
platform.openstack.lbFloatingIP	与 API 负载均衡器关联的现有浮动 IP 地址。要使用此属性，还必须定义 **platform.openstack.externalNetwork** 属性。	IP 地址，如 **128.0.0.1**。
platform.openstack.externalDNS	集群实例用于进行 DNS 解析的外部 DNS 服务器的 IP 地址。	一个 IP 地址列表作为字符串。例如，**["8.8.8.8", "192.168.1.12"]**。
platform.openstack.machinesSubnet	集群节点使用的 RHOSP 子网的 UUID。在这个子网上创建节点和虚拟 IP（VIP）端口。 **networking.machineNetwork** 中的第一个项需要和 **machinesSubnet** 的值匹配。 如果部署到自定义子网中，则无法将外部 DNS 服务器指定到 OpenShift Container Platform 安装程序。反之，把 DNS 添加到 RHOSP 的子网。	作为字符串的 UUID。例如： **fa806b2f-ac49-4bce-b9db-124bc64209bf**。

9.4.14.6. RHOSP 部署中的自定义子网

另外，您还可以在您选择的 Red Hat OpenStack Platform（RHOSP）子网中部署集群。子网的 GUID 作为 **install-config.yaml** 文件中的 **platform.openstack.machinesSubnet** 的值传递。

此子网被用作集群的主子网，在其上创建节点和端口。

在使用自定义子网运行 OpenShift Container Platform 安装程序前，请验证：

- 目标网络和子网可用。

- 目标子网上启用了 DHCP。
- 您可提供在目标网络上有创建端口权限的安装程序凭证。
- 如果您的网络配置需要一个路由器，它会在 RHOSP 中创建。有些配置依赖于路由器来转换浮动 IP 地址。
- 您的网络配置不依赖于供应商网络。不支持提供商网络。

注意

默认情况下，API VIP 使用 x.x.x.5，Ingress VIP 从网络 CIDR 块获取 x.x.x.7。要覆盖这些默认值，为 DHCP 分配池以外的 **platform.openstack.apiVIP** 和 **platform.openstack.ingressVIP** 设置值。

9.4.14.7. 使用 Kuryr 的 RHOSP 的自定义 install-config.yaml 文件示例

要使用 Kuryr SDN 而不是默认的 OpenShift SDN 部署，您必须修改 **install-config.yaml** 文件，使其包含 **Kuryr** 作为所需的 **networking.networkType**，然后执行默认的 OpenShift Container Platform SDN 安装步骤。此示例 **install-config.yaml** 展示了所有可能的 Red Hat OpenStack Platform (RHOSP) 自定义选项。

重要

此示例文件仅供参考。您必须使用安装程序来获取 **install-config.yaml** 文件。

```
apiVersion: v1
baseDomain: example.com
clusterID: os-test
controlPlane:
  name: master
  platform: {}
  replicas: 3
compute:
- name: worker
  platform:
    openstack:
      type: ml.large
  replicas: 3
metadata:
  name: example
networking:
  clusterNetwork:
  - cidr: 10.128.0.0/14
    hostPrefix: 23
  machineNetwork:
  - cidr: 10.0.0.0/16
  serviceNetwork:
  - 172.30.0.0/16 1
  networkType: Kuryr
platform:
  openstack:
    cloud: mycloud
    externalNetwork: external
```

```
    computeFlavor: m1.xlarge
    lbFloatingIP: 128.0.0.1
    trunkSupport: true 2
    octaviaSupport: true 3
pullSecret: '{"auths": ...}'
sshKey: ssh-ed25519 AAAA...
```

1 Amphora Octavia 驱动程序为每个负载均衡器创建两个端口。因此，安装程序创建的服务子网是由 **serviceNetwork** 属性值指定的 CIDR 的两倍。要防止 IP 地址冲突，则需要更大的范围。

2 3 安装程序会自动发现 **trunkSupport** 和 **octaviaSupport**，因此无需设置它们。但是，如果您的环境不满足这两个要求，Kuryr SDN 将无法正常工作。需要使用中继来把 pod 连接到 RHOSP 网络，并且需要 Octavia 来创建 OpenShift Container Platform 服务。

9.4.14.8. Kuryr 端口池

Kuryr 端口池在待机时维护多个端口，用于创建 pod。

将端口保留在待机上可最大程度缩短 pod 创建时间。如果没有端口池，Kuryr 必须明确请求在创建或删除 pod 时创建或删除端口。

Kuryr 使用的 Neutron 端口是在绑定到命名空间的子网中创建的。这些 pod 端口也作为子端口添加到 OpenShift Container Platform 集群节点的主端口。

因为 Kuryr 将每个命名空间保留在单独的子网中，所以对于每个“命名空间-worker”对都会维护一个单独的端口池。

在安装集群前，您可以在 **cluster-network-03-config.yml** 清单文件中设置以下参数来配置端口池行为：

- **enablePortPoolsPrepopulation** 参数控制池预填充，它会强制 Kuryr 在创建时（如添加新主机或创建新命名空间时）将端口添加到池中。默认值为 **false**。
- **poolMinPorts** 参数是池中保留的最少可用端口的数量。默认值为 **1**。
- **poolMaxPorts** 参数是池中保留的最大可用端口数。如果值为 **0**，会禁用上限。这是默认的设置。
  如果您的 OpenStack 端口配额较低，或者 pod 网络上的 IP 地址有限，请考虑设置此选项以确保删除不需要的端口。
- **poolBatchPorts** 参数定义一次可以创建的 Neutron 端口的最大数量。默认值为 **3**。

9.4.14.9. 在安装过程中调整 Kuryr 端口池

在安装过程中，您可以配置 Kuryr 如何管理 Red Hat OpenStack Platform (RHOSP) Neutron 端口，以控制 pod 创建的速度和效率。

先决条件

- 创建并修改 **install-config.yaml** 文件。

流程

1. 在命令行中创建清单文件：

```
$ ./openshift-install create manifests --dir <installation_directory> 1
```

1 对于 **<installation_directory>**，请指定含有集群的 **install-config.yaml** 文件的目录的名称。

2. 在 **<installation_directory>/manifests/** 目录下，创建一个名为 **cluster-network-03-config.yml** 的文件：

```
$ touch <installation_directory>/manifests/cluster-network-03-config.yml 1
```

1 对于 **<installation_directory>**，请指定包含集群的 **manifests/** 目录的目录名称。

创建该文件后，**manifests/** 目录中会包含多个网络配置文件，如下所示：

```
$ ls <installation_directory>/manifests/cluster-network-*
```

输出示例

```
cluster-network-01-crd.yml
cluster-network-02-config.yml
cluster-network-03-config.yml
```

3. 在编辑器中打开 **cluster-network-03-config.yml** 文件，并输入描述您想要的 Cluster Network Operator 配置的自定义资源(CR)：

```
$ oc edit networks.operator.openshift.io cluster
```

4. 编辑设置以满足您的要求。以下示例提供了以下文件：

```
apiVersion: operator.openshift.io/v1
kind: Network
metadata:
  name: cluster
spec:
  clusterNetwork:
  - cidr: 10.128.0.0/14
    hostPrefix: 23
  serviceNetwork:
  - 172.30.0.0/16
  defaultNetwork:
    type: Kuryr
    kuryrConfig:
      enablePortPoolsPrepopulation: false 1
      poolMinPorts: 1 2
      poolBatchPorts: 3 3
      poolMaxPorts: 5 4
      openstackServiceNetwork: 172.30.0.0/15 5
```

1 将 **enablePortPoolsPrepopulation** 的值设置为 **true** 以使 Kuryr 在创建命名空间或在集群中添加新节点后创建新 Neutron 端口。此设置引发 Neutron 端口配额，但可以缩短生成容器集所需的时间。默认值为 **false**。

2 如果池中的可用端口数量低于 **poolMinPorts** 的值，Kuryr 会为池创建新端口。默认值为 **1**。

3 **poolBatchPorts** 控制在可用端口数量低于 **poolMinPorts** 值时创建的新端口数量。默认值为 **3**。

4 如果池中的可用端口数量大于 **poolMaxPorts** 的值，Kuryr 会删除它们，直到数量与这个值匹配为止。将此值设置为 **0** 可禁用此上限，防止池缩小。默认值为 **0**。

5 **openStackServiceNetwork** 参数定义将 IP 地址分配到 RHOSP Octavia 的 LoadBalancer 的网络的 CIDR 范围。

如果此参数与 Amphora 驱动程序一起使用，则 Octavia 会为每个负载均衡器从这个网络获取两个 IP 地址：一个用于 OpenShift，另一个用于 VRRP 连接。由于这些 IP 地址分别由 OpenShift Container Platform 和 Neutron 管理，因此它们必须来自不同的池。因此，**openStackServiceNetwork** 的值必须至少是 **serviceNetwork** 值的两倍，**serviceNetwork** 的值必须与 **openStackServiceNetwork** 定义的范围完全重叠。

CNO 验证从此参数定义的范围获取的 VRRP IP 地址是否与 **serviceNetwork** 参数定义的范围不重叠。

如果没有设置此参数，CNO 将使用 **serviceNetwork** 的扩展值，它是前缀大小值减 1。

5. 保存 **cluster-network-03-config.yml** 文件，再退出文本编辑器。
6. 可选：备份 **manifests/cluster-network-03-config.yml** 文件。安装程序在创建集群时删除 **manifests/** 目录。

9.4.14.10. 为机器设置自定义子网

安装程序默认使用的 IP 范围可能与您在安装 OpenShift Container Platform 时创建的 Neutron 子网不匹配。如有必要，通过编辑安装配置文件来更新新机器的 CIDR 值。

先决条件

- 有 OpenShift Container Platform 安装程序生成的 **install-config.yaml** 文件。

流程

1. 在命令行中进入包含 **install-config.yaml** 的目录。
2. 在该目录中，运行脚本来编辑 **install-config.yaml** 文件或手动更新该文件：
   - 要使用脚本设置值，请运行：

```
$ python -c '
import yaml;
path = "install-config.yaml";
data = yaml.safe_load(open(path));
data["networking"]["machineNetwork"] = [{"cidr": "192.168.0.0/18"}]; 1
open(path, "w").write(yaml.dump(data, default_flow_style=False))'
```

1 插入一个与您指定的 Neutron 子网匹配的值，如 **192.0.2.0/24**。

- 要手动设置这个值，请打开该文件并将 **networking.machineCIDR** 的值设置为与您预期的 Neutron 子网匹配的内容。

9.4.14.11. 清空计算机器池

要进行使用您自己的基础架构的安装，请将安装配置文件中的计算机器数量设置为零。之后，您可以手动创建这些机器。

先决条件

- 有 OpenShift Container Platform 安装程序生成的 **install-config.yaml** 文件。

流程

1. 在命令行中进入包含 **install-config.yaml** 的目录。
2. 在该目录中，运行脚本来编辑 **install-config.yaml** 文件或手动更新该文件：
   - 要使用脚本设置值，请运行：

```
$ python -c '
import yaml;
path = "install-config.yaml";
data = yaml.safe_load(open(path));
data["compute"][0]["replicas"] = 0;
open(path, "w").write(yaml.dump(data, default_flow_style=False))'
```

   - 要手动设置值，打开文件并将 **compute.<first entry>.replicas** 的值设置为 **0**。

9.4.14.12. 修改网络类型

默认情况下，安装程序会选择 **OpenShiftSDN** 网络类型。要使用 Kuryr，请更改安装程序生成的安装配置文件中的值。

先决条件

- 有 OpenShift Container Platform 安装程序生成的 **install-config.yaml** 文件

流程

1. 在命令提示符中，进入包含 **install-config.yaml** 的目录。
2. 在该目录中，运行脚本来编辑 **install-config.yaml** 文件或手动更新该文件：
   - 要使用脚本设置值，请运行：

```
$ python -c '
import yaml;
path = "install-config.yaml";
data = yaml.safe_load(open(path));
data["networking"]["networkType"] = "Kuryr";
open(path, "w").write(yaml.dump(data, default_flow_style=False))'
```

   - 要手动设置这个值，打开该文件并将 **networking.networkType** 设置为 **"Kuryr"**。

9.4.15. 创建 Kubernetes 清单和 Ignition 配置文件

由于您必须修改一些集群定义文件并要手动启动集群机器，因此您必须生成 Kubernetes 清单和 Ignition 配置文件，集群需要这两项来创建其机器。

安装配置文件转换为 Kubernetes 清单。清单嵌套到 Ignition 配置文件中，稍后用于创建集群。

重要

- 安装程序生成的 Ignition 配置文件包含在 24 小时后过期的证书，然后在过期时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外情况是，您需要手动批准待处理的 **node-bootstrapper** 证书签名请求（CSR）来恢复 kubelet 证书。如需更多信息，请参阅*从过期的 control plane 证书中恢复*的文档。
- 建议您在生成 12 小时后使用 Ignition 配置文件，因为集群安装后 24 小时证书从 16 小时轮转至 22 小时。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中运行证书更新时避免安装失败。

先决条件

- 已获得 OpenShift Container Platform 安装程序。
- 已创建 **install-config.yaml** 安装配置文件。

流程

1. 切换到包含安装程序的目录，并为集群生成 Kubernetes 清单：

```
$ ./openshift-install create manifests --dir <installation_directory> 1
```

1 对于 **<installation_directory>**，请指定含有您创建的 **install-config.yaml** 文件的安装目录。

2. 删除定义 control plane 机器的 Kubernetes 清单文件以及计算机器集：

```
$ rm -f openshift/99_openshift-cluster-api_master-machines-*.yaml openshift/99_openshift-cluster-api_worker-machineset-*.yaml
```

由于您要自行创建和管理这些资源，因此不必初始化这些资源。

- 您可以使用机器 API 来保留机器集文件来创建计算机器，但您必须更新对其的引用，以匹配您的环境。

3. 检查 **<installation_directory>/manifests/cluster-scheduler-02-config.yml** Kubernetes 清单文件中的 **mastersSchedulable** 参数是否已设置为 **false**。此设置可防止在 control plane 机器上调度 pod:
   a. 打开 **<installation_directory>/manifests/cluster-scheduler-02-config.yml** 文件。
   b. 找到 **mastersSchedulable** 参数并确保它被设置为 **false**。
   c. 保存并退出文件。
4. 要创建 Ignition 配置文件，从包含安装程序的目录运行以下命令：

```
$ ./openshift-install create ignition-configs --dir <installation_directory> 1
```

1 对于 **<installation_directory>**，请指定相同的安装目录。

该目录中将生成以下文件：

```
.
├── auth
│   ├── kubeadmin-password
│   └── kubeconfig
├── bootstrap.ign
├── master.ign
├── metadata.json
└── worker.ign
```

5. 将元数据文件的 **infraID** 键导出为环境变量：

```
$ export INFRA_ID=$(jq -r .infraID metadata.json)
```

提示

从 **metadata.json** 中提取 **infraID** 键，并将其用作您创建的所有 RHOSP 资源的前缀。通过这样做，您可以避免在同一项目中进行多个部署时的名称冲突。

9.4.16. 准备 bootstrap Ignition 文件

OpenShift Container Platform 安装过程依赖于从 bootstrap Ignition 配置文件创建的 bootstrap 机器。

编辑该文件并上传该文件。然后，创建 Red Hat OpenStack Platform（RHOSP）用来下载主文件的辅助 bootstrap Ignition 配置文件。

先决条件

- 您有安装程序生成的 bootstrap Ignition 文件，即 **bootstrap.ign**。
- 安装程序元数据文件中的基础架构 ID 被设置为环境变量（**$INFRA_ID**）。
  - 如果未设置变量，请参阅 **创建 Kubernetes 清单和 Ignition 配置文件**。
- 可以使用 HTTP(S) 来存储 bootstrap ignition 文件。
  - 所记录的步骤使用 RHOSP 镜像服务（Glance），但也可以使用 RHOSP Storage 服务（Swift）、Amazon S3、内部 HTTP 服务器或临时 Nova 服务器。

流程

1. 运行以下 Python 脚本。该脚本修改 bootstrap Ignition 文件，以设置主机名，并在运行时设置 CA 证书文件：

```
import base64
import json
import os
```

```
with open('bootstrap.ign', 'r') as f:
    ignition = json.load(f)

files = ignition['storage'].get('files', [])

infra_id = os.environ.get('INFRA_ID', 'openshift').encode()
hostname_b64 = base64.standard_b64encode(infra_id + b'-bootstrap\n').decode().strip()
files.append(
{
    'path': '/etc/hostname',
    'mode': 420,
    'contents': {
        'source': 'data:text/plain;charset=utf-8;base64,' + hostname_b64
    }
})

ca_cert_path = os.environ.get('OS_CACERT', '')
if ca_cert_path:
    with open(ca_cert_path, 'r') as f:
        ca_cert = f.read().encode()
        ca_cert_b64 = base64.standard_b64encode(ca_cert).decode().strip()

    files.append(
    {
        'path': '/opt/openshift/tls/cloud-ca-cert.pem',
        'mode': 420,
        'contents': {
            'source': 'data:text/plain;charset=utf-8;base64,' + ca_cert_b64
        }
    })

ignition['storage']['files'] = files;

with open('bootstrap.ign', 'w') as f:
    json.dump(ignition, f)
```

2. 使用 RHOSP CLI，创建使用 bootstrap Ignition 文件的镜像：

```
$ openstack image create --disk-format=raw --container-format=bare --file bootstrap.ign
<image_name>
```

3. 获取镜像的详情：

```
$ openstack image show <image_name>
```

请记录 **file** 值；它需要遵循 **v2/images/<image_ID>/file** 格式。

注意

验证您创建的镜像是否活跃。

4. 检索镜像服务的公共地址：

```
$ openstack catalog show image
```

5. 将公共地址与镜像的 **file** 值合并，并在存储位置保存结果。位置遵循 **<image_service_public_URL>/v2/images/<image_ID>/file** 格式。

6. 生成身份验证令牌并保存令牌 ID:

```
$ openstack token issue -c id -f value
```

7. 将以下内容插入到名为 **$INFRA_ID-bootstrap-ignition.json** 的文件中，并编辑位置拥有者以匹配您自己的值：

```
{
  "ignition": {
    "config": {
      "merge": [{
        "source": "<storage_url>", 1
        "httpHeaders": [{
          "name": "X-Auth-Token", 2
          "value": "<token_ID>" 3
        }]
      }]
    },
    "security": {
      "tls": {
        "certificateAuthorities": [{
          "source": "data:text/plain;charset=utf-8;base64,<base64_encoded_certificate>" 4
        }]
      }
    },
    "version": "3.1.0"
  }
}
```

1. 将 **ignition.config.merge.source** 的值替换为 bootstrap Ignition 文件存储 URL。
2. 在 **httpHeaders** 中将 **name** 设置为 **"X-Auth-Token"**。
3. 在 **httpHeaders** 中将 **value** 设为您的令牌 ID。
4. 如果 bootstrap Ignition 文件服务器使用自签名证书,请包括以 base64 编码的证书。

8. 保存二级 Ignition 配置文件。

bootstrap Ignition 数据将在安装过程中传递给 RHOSP。

警告

bootstrap Ignition 文件包含敏感信息，如 **clouds.yaml** 凭证。确定您将其保存在安全的地方，并在完成安装后将其删除。

9.4.17. 在 RHOSP 上创建 control plane Ignition 配置文件

在您自己的基础架构的 Red Hat OpenStack Platform（RHOSP）上安装 OpenShift Container Platform 需要 control plane Ignition 配置文件。您必须创建多个配置文件。

注意

与 bootstrap Ignition 配置一样，您必须明确为每个 control plane 机器定义主机名。

先决条件

- 来自安装程序元数据文件中的基础架构 ID 被设置为环境变量（**$INFRA_ID**）。
  - 如果未设置变量，请参阅“创建 Kubernetes 清单和 Ignition 配置文件”。

流程

- 在命令行中运行以下 Python 脚本：

```
$ for index in $(seq 0 2); do
    MASTER_HOSTNAME="$INFRA_ID-master-$index\n"
    python -c "import base64, json, sys;
ignition = json.load(sys.stdin);
storage = ignition.get('storage', {});
files = storage.get('files', []);
files.append({'path': '/etc/hostname', 'mode': 420, 'contents': {'source':
'data:text/plain;charset=utf-8;base64,' +
base64.standard_b64encode(b'$MASTER_HOSTNAME').decode().strip(), 'verification': {}},
'filesystem': 'root'});
storage['files'] = files;
ignition['storage'] = storage
json.dump(ignition, sys.stdout)" <master.ign >"$INFRA_ID-master-$index-ignition.json"
done
```

  您现在有三个 control plane Ignition 文件：**<INFRA_ID>-master-0-ignition.json**、**<INFRA_ID>-master-1-ignition.json** 和 **<INFRA_ID>-master-2-ignition.json**。

9.4.18. 在 RHOSP 上创建网络资源

在您自己的基础架构的 Red Hat OpenStack Platform（RHOSP）安装上创建 OpenShift Container Platform 所需的网络资源。为节省时间，可以运行提供的 Ansible playbook 来生成安全组、网络、子网、路由器和端口。

先决条件

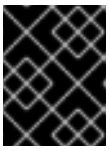
- Python 3 已安装在您的机器上。
- 您下载了"下载 playbook 依赖项"中的模块。
- 下载了"下载安装 playbook"中的 playbook。

流程

1. 可选：为 **inventory.yaml** playbook 添加一个外部网络值：

inventory.yaml Ansible playbook 中的外部网络值示例

```
...
    # The public network providing connectivity to the cluster. If not
    # provided, the cluster external connectivity must be provided in another
    # way.

    # Required for os_api_fip, os_ingress_fip, os_bootstrap_fip.
    os_external_network: 'external'
...
```



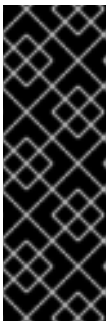
重要

如果没有为 **inventory.yaml** 文件中的 **os_external_network** 提供值，则必须确保虚拟机可以自行访问 Glance 和外部连接。

2. 可选：将外部网络和浮动 IP（FIP）地址值添加到 **inventory.yaml** playbook：

inventory.yaml Ansible playbook 中的 FIP 值示例

```
...
    # OpenShift API floating IP address. If this value is non-empty, the
    # corresponding floating IP will be attached to the Control Plane to
    # serve the OpenShift API.
    os_api_fip: '203.0.113.23'

    # OpenShift Ingress floating IP address. If this value is non-empty, the
    # corresponding floating IP will be attached to the worker nodes to serve
    # the applications.
    os_ingress_fip: '203.0.113.19'

    # If this value is non-empty, the corresponding floating IP will be
    # attached to the bootstrap machine. This is needed for collecting logs
    # in case of install failure.
    os_bootstrap_fip: '203.0.113.20'
```



重要

如果您没有为 **os_api_fip** 和 **os_ingress_fip** 定义值，则必须执行安装后的网络配置。

如果您没有为 **os_bootstrap_fip** 定义值，安装程序将无法从失败的安装中下载调试信息。

如需更多信息，请参阅"启用对环境的访问"。

3. 在命令行中，通过运行 **security-groups.yaml** playbook 来创建安全组：

```
$ ansible-playbook -i inventory.yaml security-groups.yaml
```

4. 在命令行中，通过运行 **network.yaml** playbook 来创建一个网络、子网和路由器：

```
$ ansible-playbook -i inventory.yaml network.yaml
```

5. 可选：如果要控制 Nova 服务器使用的默认解析程序，请运行 RHOSP CLI 命令：

```
$ openstack subnet set --dns-nameserver <server_1> --dns-nameserver <server_2>
"$INFRA_ID-nodes"
```

9.4.19. 在 RHOSP 上创建 bootstrap 机器

创建 bootstrap 机器，为其提供在 Red Hat OpenStack Platform（RHOSP）上运行所需的网络访问权限。红帽提供了一个 Ansible playbook，您可运行它来简化此过程。

先决条件

- 您下载了"下载 playbook 依赖项"中的模块。
- 下载了"下载安装 playbook"中的 playbook。
- **inventory.yaml**、**common.yaml** 和 **bootstrap.yaml** Ansible playbook 位于一个通用目录中。
- 安装程序创建的 **metadata.json** 文件与 Ansible playbook 位于同一个目录中。

流程

1. 在命令行中，将工作目录改为 playbook 的位置。
2. 在命令行中运行 **bootstrap.yaml** playbook：

```
$ ansible-playbook -i inventory.yaml bootstrap.yaml
```

3. bootstrap 服务器可用后，查看日志以验证是否收到 Ignition 文件：

```
$ openstack console log show "$INFRA_ID-bootstrap"
```

9.4.20. 在 RHOSP 中创建 control plane 机器

使用您生成的 Ignition 配置文件创建三台 control plane 机器。红帽提供了一个 Ansible playbook，您可运行它来简化此过程。

先决条件

- 您下载了"下载 playbook 依赖项"中的模块。
- 下载了"下载安装 playbook"中的 playbook。
- 来自安装程序元数据文件中的基础架构 ID 被设置为环境变量（**$INFRA_ID**）。
- **inventory.yaml**、**common.yaml** 和 **control-plane.yaml** Ansible playbook 位于一个通用目录中。
- 您有三个在"Creating control plane Ignition 配置文件"中创建的 Ignition 文件。

流程

1. 在命令行中，将工作目录改为 playbook 的位置。

2. 如果 control plane Ignition 配置文件尚未位于工作目录中，将其复制到其中。
3. 在命令行中运行 **control-plane.yaml** playbook：

```
$ ansible-playbook -i inventory.yaml control-plane.yaml
```

4. 运行以下命令来监控 bootstrap 过程：

```
$ openshift-install wait-for bootstrap-complete
```

您会看到确认 control plane 机器正在运行并加入集群的消息：

```
INFO API v1.14.6+f9b5405 up
INFO Waiting up to 30m0s for bootstrapping to complete...
...
INFO It is now safe to remove the bootstrap resources
```

9.4.21. 使用 CLI 登录到集群

您可以通过导出集群 **kubeconfig** 文件，以默认系统用户身份登录集群。**kubeconfig** 文件包含关于集群的信息，供 CLI 用于将客户端连接到正确集群和 API 服务器。该文件特只适用于一个特定的集群，在 OpenShift Container Platform 安装过程中创建。

先决条件

- 已部署了 OpenShift Container Platform 集群。
- 已安装 **oc** CLI。

流程

1. 导出 **kubeadmin** 凭证：

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

2. 使用导出的配置，验证能否成功运行 **oc** 命令：

```
$ oc whoami
```

输出示例

```
system:admin
```

9.4.22. 从 RHOSP 删除 bootstrap 资源

删除您不再需要的 bootstrap 资源。

先决条件

- 您下载了"下载 playbook 依赖项"中的模块。
- 下载了"下载安装 playbook"中的 playbook。
- **inventory.yaml**、**common.yaml** 和 **down-bootstrap.yaml** Ansible playbook 位于一个通用目录中。
- control plane 机器正在运行。
  - 如果您不知道机器的状态，请参阅"验证集群状态"。

流程

1. 在命令行中，将工作目录改为 playbook 的位置。
2. 在命令行中运行 **down-bootstrap.yaml** playbook：

```
$ ansible-playbook -i inventory.yaml down-bootstrap.yaml
```

bootstrap 端口、服务器和浮动 IP 地址会被删除。

警告

如果您之前没有禁用 bootstrap Ignition 文件 URL，现在需要禁用。

9.4.23. 在 RHOSP 上创建计算机器

启动 control plane 后，创建计算机器。红帽提供了一个 Ansible playbook，您可运行它来简化此过程。

先决条件

- 您下载了"下载 playbook 依赖项"中的模块。
- 下载了"下载安装 playbook"中的 playbook。
- **inventory.yaml**、**common.yaml** 和 **compute-nodes.yaml** Ansible playbook 位于一个通用目录中。
- 安装程序创建的 **metadata.json** 文件与 Ansible playbook 位于同一个目录中。
- control plane 处于活跃状态。

流程

1. 在命令行中，将工作目录改为 playbook 的位置。
2. 在命令行中运行 playbook:

```
$ ansible-playbook -i inventory.yaml compute-nodes.yaml
```

后续步骤

- 批准机器的证书签名请求。

9.4.24. 批准机器的证书签名请求

将机器添加到集群时，会为您添加的每台机器生成两个待处理证书签名请求（CSR）。您必须确认这些 CSR 已获得批准，或根据需要自行批准。客户端请求必须首先被批准，然后是服务器请求。

先决条件

- 您已将机器添加到集群中。

流程

1. 确认集群可以识别这些机器：

```
$ oc get nodes
```

输出示例

```
NAME      STATUS    ROLES   AGE  VERSION
master-0  Ready     master  63m  v1.19.0
master-1  Ready     master  63m  v1.19.0
master-2  Ready     master  64m  v1.19.0
```

输出将列出您创建的所有机器。

注意

在一些 CSR 被批准前，以上输出可能不包括计算节点（也称为 worker 节点）。

2. 检查待处理的 CSR，并确保可以看到添加到集群中的每台机器都有 **Pending** 或 **Approved** 状态的客户端请求：

```
$ oc get csr
```

输出示例

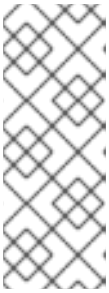
```
NAME        AGE     REQUESTOR                                                   CONDITION
csr-8b2br   15m     system:serviceaccount:openshift-machine-config-operator:node-
bootstrapper   Pending
csr-8vnps   15m     system:serviceaccount:openshift-machine-config-operator:node-
bootstrapper   Pending
...
```

在本例中，两台机器加入了集群。您可能会在列表中看到更多已批准的 CSR。

3. 如果 CSR 没有获得批准，请在所添加机器的所有待处理 CSR 都处于 **Pending** 状态后，为您的集群机器批准这些 CSR：

注意

由于 CSR 会自动轮转，因此请在将机器添加到集群后一小时内批准您的 CSR。如果没有在一小时内批准，证书将会轮转，每个节点将会存在多个证书。您必须批准所有这些证书。批准客户端 CSR 后， Kubelet 为服务证书创建辅助 CSR，这需要手动批准。然后，如果 Kubelet 请求具有相同参数的新证书，则 **machine-approver** 会自动批准后续服务证书续订请求。



注意

对于在未启用机器 API 的平台中运行的集群，如裸机和其他用户置备的基础架构，必须采用一种方法自动批准 kubelet 提供证书请求（CSR）。如果没有批准请求，则 **oc exec**、**oc rsh** 和 **oc logs** 命令将无法成功，因为 API 服务器连接到 kubelet 时需要服务证书。与 Kubelet 端点联系的任何操作都需要此证书批准。这个方法必须监视新的 CSR，确认 CSR 由 **system:node** 或 **system:admin** 组中的 **node-bootstrapper** 服务帐户提交，并确认节点的身份。

- 若要单独批准，请对每个有效的 CSR 运行以下命令：

```
$ oc adm certificate approve <csr_name> 1
```

1 **<csr_name>** 是当前 CSR 列表中 CSR 的名称。

- 要批准所有待处理的 CSR，请运行以下命令：

```
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}{{"\n"}}
{{end}}{{end}}' | xargs --no-run-if-empty oc adm certificate approve
```

注意

在有些 CSR 被批准前，一些 Operator 可能无法使用。

4. 现在，您的客户端请求已被批准，您必须查看添加到集群中的每台机器的服务器请求：

```
$ oc get csr
```

输出示例

```
NAME        AGE    REQUESTOR                                                   CONDITION
csr-bfd72   5m26s  system:node:ip-10-0-50-126.us-east-2.compute.internal
Pending
csr-c57lv   5m26s  system:node:ip-10-0-95-157.us-east-2.compute.internal
Pending
...
```

5. 如果剩余的 CSR 没有被批准，且处于 **Pending** 状态，请批准集群机器的 CSR：

- 若要单独批准，请对每个有效的 CSR 运行以下命令：

```
$ oc adm certificate approve <csr_name> 1
```

1 **<csr_name>** 是当前 CSR 列表中 CSR 的名称。

- 要批准所有待处理的 CSR，请运行以下命令：

```
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}{{"\n"}}
{{end}}{{end}}' | xargs oc adm certificate approve
```

6. 批准所有客户端和服务器 CSR 后，器将处于 **Ready** 状态。运行以下命令验证：

```
$ oc get nodes
```

输出示例

```
NAME     STATUS   ROLES   AGE  VERSION
master-0  Ready    master  73m  v1.20.0
master-1  Ready    master  73m  v1.20.0
master-2  Ready    master  74m  v1.20.0
worker-0  Ready    worker  11m  v1.20.0
worker-1  Ready    worker  11m  v1.20.0
```

注意

批准服务器 CSR 后可能需要几分钟时间让机器转换为 **Ready** 状态。

其他信息

- 如需有关 CSR 的更多信息，请参阅证书签名请求。

9.4.25. 验证安装是否成功

验证 OpenShift Container Platform 安装已完成。

先决条件

- 有安装程序（**openshift-install**）

流程

- 在命令行中运行：

```
$ openshift-install --log-level debug wait-for install-complete
```

程序输出控制台 URL 以及管理员的登录信息。

9.4.26. OpenShift Container Platform 的 Telemetry 访问

在 OpenShift Container Platform 4.6 中，默认运行的 Telemetry 服务提供有关集群健康状况和成功更新的指标，需要访问互联网。如果您的集群连接到互联网，Telemetry 会自动运行，而且集群会注册到 OpenShift Cluster Manager。

确认 OpenShift Cluster Manager 清单正确后，可以由 Telemetry 自动维护，也可以使用 OpenShift Cluster Manager 手动维护，使用订阅监控来跟踪帐户或多集群级别的 OpenShift Container Platform 订阅。

其他资源

- 有关 Telemetry 服务的更多信息，请参阅关于远程健康监控。

9.4.27. 后续步骤

- 自定义集群。
- 如果需要，您可以选择不使用远程健康报告。
- 如果您需要启用对节点端口的外部访问，请使用节点端口配置集群流量。
- 如果您没有将 RHOSP 配置为使用浮动 IP 地址接受应用程序流量，使用浮动 IP 地址配置 RHOSP 访问。

9.5. 在受限网络中的 OPENSTACK 上安装集群

在 OpenShift Container Platform 4.6 中，您可以通过创建安装发行内容的内部镜像在受限网络中的 Red Hat OpenStack Platform（RHOSP）上安装集群。

注意

只有在受限网络中安装只支持安装程序置备的安装。

先决条件

- 在镜像主机上创建镜像 registry，并获取您的 OpenShift Container Platform 版本的 **imageContentSources** 数据。

  重要

  由于安装介质位于堡垒主机上，因此请使用该计算机完成所有安装步骤。

- 查看有关 OpenShift Container Platform 安装和更新流程 的详细信息。
  - 通过咨询架构文档的可用平台列表来验证 OpenShift Container Platform 4.6 是否与您的 RHOSP 版本兼容。您还可以查看 OpenShift Container Platform 在 RHOSP 中的支持 来比较不同版本的平台支持。
- 验证您的网络配置不依赖于供应商网络。不支持提供商网络。
- 在 RHOSP 中启用了元数据服务。

9.5.1. 关于在受限网络中安装

在 OpenShift Container Platform 4.6 中，可以执行不需要有效的互联网连接来获取软件组件的安装。受限网络安装可使用安装程序置备的基础架构或用户置备的基础架构完成，具体取决于您要安装集群的云平台。

如果选择在云平台中执行受限网络安装，仍然需要访问其云 API。有些云功能，比如 Amazon Web Service 的 Route 53 DNS 和 IAM 服务，需要访问互联网。根据您的网络，在裸机硬件或 VMware vSphere 上安装时可能需要较少的互联网访问。

要完成受限网络安装，您必须创建一个 registry，镜像 OpenShift Container Platform registry 的内容并包含其安装介质。您可以在堡垒主机上创建此镜像，该主机可同时访问互联网和您的封闭网络，也可以使用满足您的限制条件的其他方法。

9.5.1.1. 其他限制

受限网络中的集群还有以下额外限制：

- **ClusterVersion** 状态包含一个 **Unable to retrieve available updates** 错误。
- 默认情况下，您无法使用 Developer Catalog 的内容，因为您无法访问所需的镜像流标签。

9.5.2. 在 RHOSP 上安装 OpenShift Container Platform 的资源指南

您的 Red Hat OpenStack Platform（RHOSP）配额需要满足以下条件才支持 OpenShift Container Platform 安装：

表 9.25. RHOSP 上默认 OpenShift Container Platform 集群的建议资源

资源	值
浮动 IP 地址	3
端口	15
路由器	1
子网	1
RAM	112 GB
vCPUs	28
卷存储	275 GB
实例	7
安全组	3
安全组规则	60

集群或许能使用少于推荐数量的资源来运作，但其性能无法保证。

重要

如果 RHOSP 对象存储（Swift）可用，并由具有 **swiftoperator** 角色的用户帐户执行，它会作为 OpenShift Container Platform 镜像 registry 的默认后端。在这种情况下，卷存储需要有 175GB。根据镜像 registry 的大小，Swift 空间要求会有所不同。

注意

默认情况下，您的安全组和安全组规则配额可能较低。如果遇到问题，请以 admin 的身份运行 **openstack quota set --secgroups 3 --secgroup-rules 60 <project>** 来提高配额。

OpenShift Container Platform 部署由 control plane 机器、计算机器和 bootstrap 机器组成。

9.5.2.1. control plane 机器

默认情况下，OpenShift Container Platform 安装过程会创建三台 control plane 机器。

每台机器都需要：

- 来自 RHOSP 配额的实例
- 来自 RHOSP 配额的端口
- 至少 16 GB 内存、4 个 vCPU 和 100 GB 存储空间的类别

9.5.2.2. 计算机器

默认情况下，OpenShift Container Platform 安装过程会创建三台计算机器。

每台机器都需要：

- 来自 RHOSP 配额的实例
- 来自 RHOSP 配额的端口
- 至少有 8 GB 内存、2 个 vCPU 和 100 GB 存储空间的类别

提示

计算机器托管您在 OpenShift Container Platform 上运行的应用程序；运行数量应尽可能多。

9.5.2.3. bootstrap 机器

在安装时，会临时置备 bootstrap 机器来支持 control plane。生产控制平面就绪后，bootstrap 机器会被取消置备。

bootstrap 机器需要：

- 来自 RHOSP 配额的实例
- 来自 RHOSP 配额的端口
- 至少 16 GB 内存、4 个 vCPU 和 100 GB 存储空间的类别

9.5.3. OpenShift Container Platform 的互联网访问

在 OpenShift Container Platform 4.6 中，您需要访问互联网来获得用来安装集群的镜像。

您必须具有以下互联网访问权限：

- 访问 OpenShift Cluster Manager 以下载安装程序并执行订阅管理。如果集群可以访问互联网，并且没有禁用 Telemetry，该服务会自动授权您的集群。
- 访问 Quay.io，以获取安装集群所需的软件包。
- 获取执行集群更新所需的软件包。

重要

如果您的集群无法直接访问互联网，则可以在置备的某些类基础架构上执行受限网络安装。在此过程中，您要下载所需的内容，并使用它在镜像 registry（mirror registry） 中填充安装集群并生成安装程序所需的软件包。对于某些安装类型，集群要安装到的环境不需要访问互联网。在更新集群之前，要更新 registry 镜像系统中的内容。

9.5.4. 在 RHOSP 上启用 Swift

Swift 由具有 **swiftoperator** 角色的用户帐户操控。在运行安装程序前，将该角色添加到帐户。

重要

如果 Red Hat OpenStack Platform（RHOSP） 对象存储服务 （通常称为 Swift）可用，OpenShift Container Platform 会使用它作为镜像 registry 存储。如果无法使用，安装程序将依赖于 RHOSP 快存储服务，通常称为 Cinder 。

如果 Swift 存在且您想要使用 Swift，则必须启用对其的访问。如果不存在，或者您不想使用它，请跳过这个部分。

先决条件

- 在目标环境中具有 RHOSP 管理员帐户
- 已安装 Swift 服务。
- 在Ceph RGW 上启用了**account in url** 选项。

流程

在 RHOSP 上启用 Swift：

1. 在 RHOSP CLI 中以管理员身份，将 **swiftoperator** 角色添加到要访问 Swift 的帐户：

```
$ openstack role add --user <user> --project <project> swiftoperator
```

您的 RHOSP 部署现可以使用 Swift 用于镜像 registry。

9.5.5. 为安装程序定义参数

OpenShift Container Platform 安装程序依赖于一个名为 **clouds.yaml** 的文件。该文件描述了 Red Hat OpenStack Platform (RHOSP) 配置参数，包括项目名称、登录信息和授权服务 URL。

流程

1. 创建 **clouds.yaml** 文件：

   - 如果您的 RHOSP 发行版包含 Horizon web UI，请在该 UI 中生成 **clouds.yaml** 文件。

     重要

     请记住在 **auth** 字段中添加密码。您也可以把 secret 保存在 **clouds.yaml** 以外的一个独立的文件中。

   - 如果您的 RHOSP 发行版不包含 Horizon Web UI，或者您不想使用 Horizon，请自行创建该文件。如需有关 **clouds.yaml** 的详细信息，请参阅 RHOSP 文档中的配置文件。

     ```
     clouds:
       shiftstack:
         auth:
           auth_url: http://10.10.14.42:5000/v3
           project_name: shiftstack
           username: shiftstack_user
           password: XXX
           user_domain_name: Default
           project_domain_name: Default
       dev-env:
         region_name: RegionOne
         auth:
           username: 'devuser'
           password: XXX
           project_name: 'devonly'
           auth_url: 'https://10.10.14.22:5001/v2.0'
     ```

2. 如果您的 RHOSP 安装使用自签名证书颁发机构 (CA) 证书进行端点身份验证：

   a. 将 CA 文件复制到您的机器中。

   b. 将机器添加到证书颁发机构信任捆绑包中：

      ```
      $ sudo cp ca.crt.pem /etc/pki/ca-trust/source/anchors/
      ```

   c. 更新信任捆绑包：

      ```
      $ sudo update-ca-trust extract
      ```

   d. 将 **cacerts** 键添加到 **clouds.yaml** 文件。该值必须是到 CA 证书的绝对路径，则其可以被非根用户访问：

      ```
      clouds:
        shiftstack:
          ...
          cacert: "/etc/pki/ca-trust/source/anchors/ca.crt.pem"
      ```

提示

使用自定义 CA 证书运行安装程序后，您可以通过编辑 **cloud-provider-config** keymap 中的**ca-cert.pem** 键的值来更新证书。在命令行中运行：

```
$ oc edit configmap -n openshift-config cloud-provider-config
```

3. 将 **clouds.yaml** 文件放在以下位置之一：

   a. **OS_CLIENT_CONFIG_FILE** 环境变量的值

   b. 当前目录

   c. 特定于 Unix 的用户配置目录，如 **~/.config/openstack/clouds.yaml**

   d. 特定于 Unix 的站点配置目录，如 **/etc/openstack/clouds.yaml**
   安装程序会按照以上顺序搜索 **clouds.yaml**。

9.5.6. 为受限网络安装创建 RHCOS 镜像

下载 Red Hat Enterprise Linux CoreOS（RHCOS）镜像，以便在受限网络 Red Hat OpenStack Platform（RHOSP）环境中安装 OpenShift Container Platform。

先决条件

- 获取 OpenShift Container Platform 安装程序。对于受限网络安装，该程序位于您的镜像 registry 主机上。

流程

1. 登录到红帽客户门户网站的产品下载页。
2. 在 **Version** 下，为 RHEL 8 选择 OpenShift Container Platform 4.6 的最新发行版本。

重要

RHCOS 镜像可能不会随着 OpenShift Container Platform 的每一发行版本都有改变。您必须下载最高版本的镜像，其版本号应小于或等于您安装的 OpenShift Container Platform 版本。如果可用，请使用与 OpenShift Container Platform 版本匹配的镜像版本。

3. 下载 **Red Hat Enterprise Linux CoreOS（RHCOS）- OpenStack Image（QCOW）**镜像。
4. 解压镜像。

注意

您必须解压镜像，然后集群才能使用它。下载的文件名可能不包含压缩扩展名，如 **.gz** 或 **.tgz**。要找出是否或者如何压缩文件，请在命令行中输入：

```
$ file <name_of_downloaded_file>
```

5. 将您解压缩的镜像上传到堡垒服务器可访问的位置，如 Glance。例如：

```
$ openstack image create --file rhcos-44.81.202003110027-0-openstack.x86_64.qcow2 --
disk-format qcow2 rhcos-${RHCOS_VERSION}
```

重要

根据您的 RHOSP 环境，可能需要使用 **.raw** 或 **.qcow2** 格式下载镜像。如果使用 Ceph，则必须使用 **.raw** 格式。

警告

如果安装程序发现多个同名的镜像，它会随机选择其中之一。为避免这种行为，请在 RHOSP 中为资源创建唯一名称。

该镜像现在可用于受限安装。记录 OpenShift Container Platform 部署中使用的镜像名称或位置。

9.5.7. 创建安装配置文件

您可以自定义在 Red Hat OpenStack Platform（RHOSP）上安装的 OpenShift Container Platform 集群。

先决条件

- 获取 OpenShift Container Platform 安装程序以及集群的 pull secret。对于受限网络安装，这些文件位于您的堡垒主机上。
- 具有创建镜像容器镜像仓库（registry）时生成的 **imageContentSources** 值。
- 获取您的镜像 registry 的证书内容。
- 检索 Red Hat Enterprise Linux CoreOS（RHCOS）镜像，并将其上传到可访问的位置。

流程

1. 创建 **install-config.yaml** 文件。
   a. 更改到包含安装程序的目录，再运行以下命令：

```
$ ./openshift-install create install-config --dir <installation_directory> 1
```

1 对于 **<installation_directory>**，请指定用于保存安装程序所创建的文件的目录名称。

重要

指定一个空目录。一些安装信息，如 bootstrap X.509 证书，有较短的过期间隔，因此不要重复使用安装目录。如果要重复使用另一个集群安装中的个别文件，可以将其复制到您的目录中。但是，一些安装数据的文件名可能会在发行版本之间有所改变。从 OpenShift Container Platform 老版本中复制安装文件时要格外小心。

b. 在提示符处，提供您的云的配置详情：

i. 可选：选择用来访问集群机器的 SSH 密钥。

注意

对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 **ssh-agent** 进程使用的 SSH 密钥。

ii. 选择 **openstack** 作为目标平台。

iii. 指定用于安装集群的 Red Hat OpenStack Platform (RHOSP) 外部网络名称。

iv. 指定用于从外部访问 OpenShift API 的浮动 IP 地址。

v. 指定至少有 16 GB RAM 用于 control plane 节点，以及计算节点的 8 GB RAM。

vi. 选择集群要部署到的基域。所有 DNS 记录都将是这个基域的子域，并包含集群名称。

vii. 为集群输入一个名称。名称不能多于 14 个字符。

viii. 粘贴 Red Hat OpenShift Cluster Manager 中的 pull secret 。

2. 在 **install-config.yaml** 文件中，将 **platform.openstack.clusterOSImage** 的值设置为镜像位置或名称。例如：

```
platform:
  openstack:
    clusterOSImage: http://mirror.example.com/images/rhcos-43.81.201912131630.0-
openstack.x86_64.qcow2.gz?
sha256=ffebbd68e8a1f2a245ca19522c16c86f67f9ac8e4e0c1f0a812b068b16f7265d
```

3. 编辑 **install-config.yaml** 文件，以提供在受限网络中安装所需的其他信息。

a. 更新 **pullSecret** 值，使其包含 registry 的身份验证信息：

```
pullSecret: '{"auths":{"<mirror_host_name>:5000": {"auth": "<credentials>","email":
"you@example.com"}}}'
```

对于 **<mirror_host_name>**，请指定您在镜像 registry 证书中指定的 registry 域名；对于 **<credentials>**，请指定您的镜像 registry 的 base64 编码用户名和密码。

b. 添加 **additionalTrustBundle** 参数和值。

```
additionalTrustBundle: |
  -----BEGIN CERTIFICATE-----

ZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZ
  -----END CERTIFICATE-----
```

该值必须是您用于镜像 registry 的证书文件内容，可以是现有的可信证书颁发机构或您为镜像 registry 生成的自签名证书。

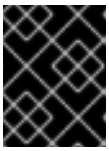
c. 添加镜像内容资源，如下例所示：

```
imageContentSources:
- mirrors:
  - <mirror_host_name>:5000/<repo_name>/release
  source: quay.example.com/openshift-release-dev/ocp-release
- mirrors:
  - <mirror_host_name>:5000/<repo_name>/release
  source: registry.example.com/ocp/release
```

要完成这些值，请使用您在创建镜像容器镜像仓库（registry）时记录的 **imageContentSources**。

4. 对需要的 **install-config.yaml** 文件做任何其他修改。您可以在**安装配置参数**部分中找到有关可用参数的更多信息。

5. 备份 **install-config.yaml** 文件，以便用于安装多个集群。



重要

install-config.yaml 文件会在安装过程中消耗掉。如果要重复使用此文件，必须现在备份。

9.5.7.1. 在安装过程中配置集群范围代理

生产环境可能会拒绝直接访问互联网，而是提供 HTTP 或 HTTPS 代理。您可以通过在 **install-config.yaml** 文件中配置代理设置，将新的 OpenShift Container Platform 集群配置为使用代理。

先决条件

- 您有一个现有的 **install-config.yaml** 文件。
- 您检查了集群需要访问的站点，并决定是否需要绕过代理。默认情况下代理所有集群出口流量，包括对托管云供应商 API 的调用。您需要将站点添加到 **Proxy** 对象的 **spec.noProxy** 字段来绕过代理。

注意

Proxy 对象 **status.noProxy** 字段使用安装配置中的 **networking.machineNetwork[].cidr**、**networking.clusterNetwork[].cidr** 和 **networking.serviceNetwork[]** 字段的值填充。

对于在 Amazon Web Services(AWS)、Google Cloud Platform(GCP)、Microsoft Azure 和 Red Hat OpenStack Platform(RHOSP)上安装, **Proxy** 对象 **status.noProxy** 字段也会使用实例元数据端点填充(**169.254.169.254**)。

流程

1. 编辑 **install-config.yaml** 文件并添加代理设置。例如：

```
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: example.com 3
```

```
additionalTrustBundle: | 4
  -----BEGIN CERTIFICATE-----
  <MY_TRUSTED_CA_CERT>
  -----END CERTIFICATE-----
...
```

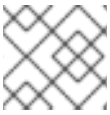
1. 用于创建集群外 HTTP 连接的代理 URL。URL 必须是 **http**。
2. 用于创建集群外 HTTPS 连接的代理 URL。
3. 要排除在代理中的目标域名、IP 地址或其他网络 CIDR 的逗号分隔列表。在域前面加 **.** 来仅匹配子域。例如： **.y.com** 匹配 **x.y.com**，但不匹配 **y.com**。使用 ***** 绕过所有目的地的代理。
4. 如果提供，安装程序会在 **openshift-config 命名空间中生成名为 user-ca- bundle** 的配置映射来保存额外的 CA 证书。如果您提供 **additionalTrustBundle** 和至少一个代理设置，则 **Proxy** 对象会被配置为引用 **trustedCA** 字段中的 **user-ca-bundle** 配置映射。然后，Cluster Network Operator 会创建一个 **trusted-ca-bundle** 配置映射，该配置映射将为 **trustedCA** 参数指定的内容与 RHCOS 信任捆绑包合并。 **additionalTrustBundle** 字段是必需的，除非代理的身份证书由来自 RHCOS 信任捆绑包的颁发机构签名。

注意

安装程序不支持代理的 **readinessEndpoints** 字段。

2. 保存该文件，并在安装 OpenShift Container Platform 时引用。

安装程序会创建一个名为 **cluster** 的集群范围代理，该代理使用提供的 **install-config.yaml** 文件中的代理设置。如果没有提供代理设置，仍然会创建一个 **cluster Proxy** 对象，但它会有一个空 **spec**。

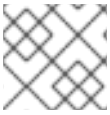


注意

只支持名为 **cluster** 的 **Proxy** 对象，且无法创建额外的代理。

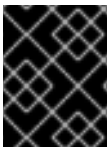
9.5.7.2. 安装配置参数

在部署 OpenShift Container Platform 集群前，您可以提供参数值，以描述托管集群的云平台的帐户并选择性地自定义集群平台。在创建 **install-config.yaml** 安装配置文件时，您可以通过命令行来提供所需的参数的值。如果要自定义集群，可以修改 **install-config.yaml** 文件来提供关于平台的更多信息。



注意

安装之后，您无法修改 **install-config.yaml** 文件中的这些参数。



重要

openshift-install 命令不验证参数的字段名称。如果指定了不正确的名称，则不会创建相关的文件或对象，且不会报告错误。确保所有指定的参数的字段名称都正确。

9.5.7.2.1. 所需的配置参数

下表描述了所需的安装配置参数：

表 9.26. 所需的参数

参数	描述	值
apiVersion	**install-config.yaml** 内容的 API 版本。当前版本是 **v1**。安装程序还可能支持旧的 API 版本。	字符串
baseDomain	云供应商的基域。此基础域用于创建到 OpenShift Container Platform 集群组件的路由。集群的完整 DNS 名称是 **baseDomain** 和 **metadata.name** 参数值的组合，其格式为 **<metadata.name>.<baseDomain>**。	完全限定域名或子域名，如 **example.com**。
metadata	Kubernetes 资源 **ObjectMeta**，其中只消耗 **name** 参数。	对象
metadata.name	集群的名称。集群的 DNS 记录是 **{{.metadata.name}}.{{.baseDomain}}** 的子域。	小写字母,连字符(-)和句点(.)的字符串，如 **dev**。该字符串长度必须为 14 个字符或更少。
platform	执行安装的具体平台配置：**aws**、**baremetal**、**azure**、**openstack**、**ovirt**、**vsphere**。有关 **platform.<platform>** 参数的额外信息，请参考下表来了解您的具体平台。	对象

pullSecret	从 Red Hat OpenShift Cluster Manager 获取 pull secret，验证从 Quay.io 等服务中下载 OpenShift Container Platform 组件的容器镜像。	{
		"auths":{
		"cloud.openshift.com":{
		"auth":"b3Blb=",
		"email":"you@example.com"
		},
		"quay.io":{
		"auth":"b3Blb=",
		"email":"you@example.com"
		}
		}
		}

9.5.7.2.2. 网络配置参数

您可以根据现有网络基础架构的要求自定义安装配置。例如，您可以扩展集群网络的 IP 地址块，或者提供不同于默认值的不同 IP 地址块。

只支持 IPv4 地址。

表 9.27. 网络参数

参数	描述	值
networking	集群网络的配置。	对象 **注意** 您不能在安装后修改 **networking** 对象指定的参数。
networking.networkType	要安装的集群网络供应商 Container Network Interface（CNI）插件。	**OpenShiftSDN** 或 **OVNKubernetes**。默认值为 **OpenShiftSDN**。
networking.clusterNetwork	pod 的 IP 地址块。 默认值为 **10.128.0.0/14**，主机前缀为 **/23**。 如果您指定多个 IP 地址块，则块不得互相重叠。	一个对象数组。例如： networking: clusterNetwork: - cidr: 10.128.0.0/14 hostPrefix: 23
networking.clusterNetwork.cidr	使用 **networking.clusterNetwork** 时需要此项。IP 地址块。 一个 IPv4 网络。	使用 CIDR 形式的 IP 地址块。IPv4 块的前缀长度介于 **0** 到 **32** 之间。
networking.clusterNetwork.hostPrefix	分配给每个单独节点的子网前缀长度。例如，如果 **hostPrefix** 设为 **23**，则每个节点从所给的 **cidr** 中分配一个 **/23** 子网。**hostPrefix** 值 **23** 提供 510（2^(32 - 23)- 2）个 pod IP 地址。	子网前缀。 默认值为 **23**。
networking.serviceNetwork	服务的 IP 地址块。默认值为 **172.30.0.0/16**。 OpenShift SDN 和 OVN-Kubernetes 网络供应商只支持服务网络的一个 IP 地址块。	CIDR 格式具有 IP 地址块的数组。例如： networking: serviceNetwork: - 172.30.0.0/16
networking.machineNetwork	机器的 IP 地址块。 如果您指定多个 IP 地址块，则块不得互相重叠。	一个对象数组。例如： networking: machineNetwork: - cidr: 10.0.0.0/16

参数	描述	值
networking.machineNetwork.cidr	使用 **networking.machineNetwork** 时需要。IP 地址块。libvirt 以外的所有平台的默认值为 **10.0.0.0/16**。对于 libvirt，默认值为 **192.168.126.0/24**。	CIDR 表示法中的 IP 网络块。 例如： **10.0.0.0/16**。  **注意** 将 **networking.machineNetwork** 设置为与首选 NIC 所在的 CIDR 匹配。

9.5.7.2.3. 可选配置参数

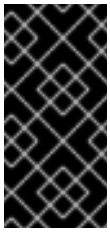
下表描述了可选安装配置参数：

表 9.28. 可选参数

参数	描述	值
additionalTrustBundle	添加到节点可信证书存储中的 PEM 编码 X.509 证书捆绑包。配置了代理时，也可以使用这个信任捆绑包。	字符串
compute	组成计算节点的机器的配置。	machine-pool 对象的数组。详情请查看以下"Machine-pool"表。
compute.architecture	决定池中机器的指令集合架构。目前不支持异构集群，因此所有池都必须指定相同的架构。有效值为 **amd64** （默认值）。	字符串
compute.hyperthreading	是否在计算机器上启用或禁用并发多线程或**超线程**。默认情况下，启用并发多线程以提高机器内核的性能。  **重要** 如果禁用并发多线程，请确保在容量规划时考虑到机器性能可能会显著降低的问题。	**Enabled** 或 **Disabled**
compute.name	使用 **compute** 时需要此值。机器池的名称。	**worker**

参数	描述	值
compute.platform	使用 **compute** 时需要此值。使用此参数指定托管 worker 机器的云供应商。此参数值必须与 **controlPlane.platform** 参数值匹配。	**aws**、**azure**、**gcp**、**openstack**、**ovirt**、**vsphere** 或 **{}**
compute.replicas	要置备的计算机器数量，也称为 worker 机器。	大于或等于 **2** 的正整数。默认值为 **3**。
controlPlane	组成 control plane 的机器的配置。	**MachinePool** 对象的数组。详情请查看以下"Machine-pool"表。
controlPlane.architecture	决定池中机器的指令集合架构。目前不支持异构集群，因此所有池都必须指定相同的架构。有效值为 **amd64** （默认值）。	字符串
controlPlane.hyperthreading	是否在 control plane 机器上启用或禁用并发多线程或**超线程**。默认情况下，启用并发多线程以提高机器内核的性能。  **重要** 如果禁用并发多线程，请确保在容量规划时考虑到机器性能可能会显著降低的问题。	**Enabled** 或 **Disabled**
controlPlane.name	使用 **controlPlane** 时需要。机器池的名称。	**master**
controlPlane.platform	使用 **controlPlane** 时需要。使用此参数指定托管 control plane 机器的云供应商。此参数值必须与 **compute.platform** 参数值匹配。	**aws**、**azure**、**gcp**、**openstack**、**ovirt**、**vsphere** 或 **{}**
controlPlane.replicas	要置备的 control plane 机器数量。	唯一支持的值是 **3**，它是默认值。

参数	描述	值
credentialsMode	Cloud Credential Operator（CCO）模式。如果没有指定任何模式，CCO 会动态地尝试决定提供的凭证的功能，在支持多个模式的平台上使用 mint 模式。  **注意** 不是所有 CCO 模式都支持所有云供应商。如需有关 CCO 模式的更多信息，请参阅*Red Hat Operator 参考指南*内容中的 *Cloud Credential Operator* 条目。	**Mint**、**Passthrough**、**Manual** 或空字符串(**""**)。
fips	启用或禁用 FIPS 模式。默认为 **false**（禁用）。如果启用了 FIPS 模式，运行 OpenShift Container Platform 的 Red Hat Enterprise Linux CoreOS (RHCOS) 机器会绕过默认的 Kubernetes 加密套件，并使用由 RHCOS 提供的加密模块。  **重要** 只有在 **x86_64** 架构中的 OpenShift Container Platform 部署支持 FIPS 验证的/Modules in Process 加密库。  **注意** 如果使用 Azure File 存储，则无法启用 FIPS 模式。	**false** 或 **true**
imageContentSources	release-image 内容的源和仓库。	对象数组。包括一个 **source** 以及可选的 **mirrors**，如下表所示。
imageContentSources.source	使用 **imageContentSources** 时需要。指定用户在镜像拉取规格中引用的仓库。	字符串

参数	描述	值
imageContentSources.mirrors	指定可能还包含同一镜像的一个或多个仓库。	字符串数组
publish	如何发布或公开集群的面向用户的端点，如 Kubernetes API、OpenShift 路由。	**Internal** 或 **External**。默认值为 **External**。 在非云平台上不支持将此字段设置为 **Internal**。  **重要** 如果将字段的值设为 **Internal**，集群将无法运行。如需更多信息，请参阅 BZ#1953035。

sshKey	用于验证集群机器访问的 SSH 密钥或密钥。
	
	 
	注意
	
	对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 **ssh-agent** 进程使用的 SSH 密钥。
		一个或多个密钥。例如：
		
		`sshKey:`
		`  <key1>`
		`  <key2>`
		`  <key3>`

9.5.7.2.4. 其他 Red Hat OpenStack Platform（RHOSP）配置参数

下表描述了其他 RHOSP 配置参数：

表 9.29. 其他 RHOSP 参数

参数	描述	值
compute.platform.openstack.rootVolume.size	对于计算机器，以 GB 为单位表示的根卷大小。如果您不设置这个值，机器将使用临时存储。	整数，如 **30**。
compute.platform.openstack.rootVolume.type	对于计算机器，根卷的类型。	字符串，如 **performance**。

参数	描述	值
controlPlane.platform.openstack.rootVolume.size	对于 control plane 机器，以 GB 为单位表示的根卷大小。如果您不设置这个值，机器将使用临时存储。	整数，如 **30**。
controlPlane.platform.openstack.rootVolume.type	对于 control plane 机器，根卷的类型。	字符串，如 **performance**。
platform.openstack.cloud	要使用的 RHOSP 云的名称，来自于 **clouds.yaml** 文件中的云列表。	字符串，如 **MyCloud**。
platform.openstack.externalNetwork	用于安装的 RHOSP 外部网络名称。	字符串，如 **external**。
platform.openstack.computeFlavor	用于 control plane 和计算机器的 RHOSP 类别。	字符串，如 **m1.xlarge**。

9.5.7.2.5. 可选 RHOSP 配置参数

下表描述了可选 RHOSP 配置参数：

表 9.30. 可选的 RHOSP 参数

参数	描述	值
compute.platform.openstack.additionalNetworkIDs	与计算机器关联的其他网络。不能为额外网络创建允许的地址对。	一个或多个 UUID 列表作为字符串。例如：**fa806b2f-ac49-4bce-b9db-124bc64209bf**。
compute.platform.openstack.additionalSecurityGroupIDs	与计算机器关联的其他安全组。	一个或多个 UUID 列表作为字符串。例如：**7ee219f3-d2e9-48a1-96c2-e7429f1b0da7**。

参数	描述	值
compute.platform.openstack.zones	RHOSP Compute（Nova）可用区（AZ）在其中安装机器。如果没有设置此参数，安装程序会依赖于配置了 RHOSP 管理员的 Nova 的默认设置。 在使用 Kuryr 的集群上，RHOSP Octavia 不支持可用域。负载均衡器，如果您使用 Amphora 供应商驱动程序，则依赖 Amphora 虚拟机的 OpenShift Container Platform 服务不会根据此属性的值创建。	字符串列表。例如：**["zone-1", "zone-2"]**。
controlPlane.platform.openstack.additionalNetworkIDs	与 control plane 机器关联的额外网络。不能为额外网络创建允许的地址对。	一个或多个 UUID 列表作为字符串。例如：**fa806b2f-ac49-4bce-b9db-124bc64209bf**。
controlPlane.platform.openstack.additionalSecurityGroupIDs	与 control plane 机器关联的其他安全组。	一个或多个 UUID 列表作为字符串。例如：**7ee219f3-d2e9-48a1-96c2-e7429f1b0da7**。
controlPlane.platform.openstack.zones	RHOSP Compute（Nova）可用区（AZ）在其中安装机器。如果没有设置此参数，安装程序会依赖于配置了 RHOSP 管理员的 Nova 的默认设置。 在使用 Kuryr 的集群上，RHOSP Octavia 不支持可用域。负载均衡器，如果您使用 Amphora 供应商驱动程序，则依赖 Amphora 虚拟机的 OpenShift Container Platform 服务不会根据此属性的值创建。	字符串列表。例如：**["zone-1", "zone-2"]**。

参数	描述	值
platform.openstack.clusterOSImage	安装程序从中下载 RHCOS 镜像的位置。 您必须设置此参数以便在受限网络中执行安装。	HTTP 或 HTTPS URL，可选使用 SHA-256 checksum。 例如：**http://mirror.example.com/images/rhcos-43.81.201912131630.0-openstack.x86_64.qcow2.gz?sha256=ffebbd68e8a1f2a245ca19522c16c86f67f9ac8e4e0c1f0a812b068b16f7265d**。该值也可以是现有 Glance 镜像的名称，如 **my-rhcos**。

platform.openstack.defaultMachinePlatform	默认机器池平台配置。	<pre>{
		  "type": "ml.large",
		  "rootVolume": {
		  "size": 30,
		  "type": "performance"
		  }
		}</pre>

platform.openstack.ingressFloatingIP	与 Ingress 端口关联的现有浮动 IP 地址。要使用此属性，还必须定义 **platform.openstack.externalNetwork** 属性。	IP 地址，如 **128.0.0.1**。
platform.openstack.lbFloatingIP	与 API 负载均衡器关联的现有浮动 IP 地址。要使用此属性，还必须定义 **platform.openstack.externalNetwork** 属性。	IP 地址，如 **128.0.0.1**。
platform.openstack.externalDNS	集群实例用于进行 DNS 解析的外部 DNS 服务器的 IP 地址。	一个 IP 地址列表作为字符串。例如，**["8.8.8.8", "192.168.1.12"]**。

参数	描述	值
platform.openstack.machinesSubnet	集群节点使用的 RHOSP 子网的 UUID。在这个子网上创建节点和虚拟 IP（VIP）端口。 **networking.machineNetwork** 中的第一个项需要和 **machinesSubnet** 的值匹配。 如果部署到自定义子网中，则无法将外部 DNS 服务器指定到 OpenShift Container Platform 安装程序。反之，把 DNS 添加到 RHOSP 的子网。	作为字符串的 UUID。例如： **fa806b2f-ac49-4bce-b9db-124bc64209bf**。

9.5.7.3. 受限 OpenStack 安装的自定义 install-config.yaml 文件示例

此示例 **install-config.yaml** 展示了所有可能的 Red Hat OpenStack Platform (RHOSP) 自定义选项。

重要

此示例文件仅供参考。您必须使用安装程序来获取 **install-config.yaml** 文件。

```
apiVersion: v1
baseDomain: example.com
clusterID: os-test
controlPlane:
  name: master
  platform: {}
  replicas: 3
compute:
- name: worker
  platform:
    openstack:
      type: ml.large
  replicas: 3
metadata:
  name: example
networking:
  clusterNetwork:
  - cidr: 10.128.0.0/14
    hostPrefix: 23
  machineCIDR: 10.0.0.0/16
  serviceNetwork:
  - 172.30.0.0/16
  networkType: OpenShiftSDN
platform:
  openstack:
    region: region1
    cloud: mycloud
    externalNetwork: external
```

```
    computeFlavor: m1.xlarge
    lbFloatingIP: 128.0.0.1
fips: false
pullSecret: '{"auths": ...}'
sshKey: ssh-ed25519 AAAA...
additionalTrustBundle: |

  -----BEGIN CERTIFICATE-----

  ZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZ

  -----END CERTIFICATE-----

imageContentSources:
- mirrors:
  - <mirror_registry>/<repo_name>/release
  source: quay.io/openshift-release-dev/ocp-release
- mirrors:
  - <mirror_registry>/<repo_name>/release
  source: quay.io/openshift-release-dev/ocp-v4.0-art-dev
```

9.5.8. 设置计算机器关联性

另外，您还可以在安装过程中为计算机器设置关联性策略。默认情况下，安装程序不会为计算机器选择关联性策略。

您还可以在安装后创建使用特定 RHOSP 服务器组的机器集。

注意

control plane 机器使用 **soft-anti-affinity** 策略创建。

提示

您可以在 RHOSP 文档中了解更多有关 RHOSP 实例调度和放置 的信息。

先决条件

- 创建 **install-config.yaml** 文件并完成对其所做的任何修改。

流程

1. 使用 RHOSP 命令行界面，为您的计算机器创建服务器组。例如：

```
$ openstack \
    --os-compute-api-version=2.15 \
    server group create \
    --policy anti-affinity \
    my-openshift-worker-group
```

   如需更多信息，请参阅**服务器组 create** 命令文档。

2. 进入包含安装程序的目录并创建清单：

```
$ ./openshift-install create manifests --dir=<installation_directory>
```

其中：

installation_directory

指定包含集群的 **install-config.yaml** 文件的目录名称。

3. 打开 **manifests/99_openshift-cluster-api_worker-machineset-0.yaml**，这是 **MachineSet** 定义文件。

4. 将属性 **serverGroupID** 添加到 **spec.template.spec.providerSpec.value** 属性下的定义中。例如：

```
apiVersion: machine.openshift.io/v1beta1
kind: MachineSet
metadata:
  labels:
    machine.openshift.io/cluster-api-cluster: <infrastructure_ID>
    machine.openshift.io/cluster-api-machine-role: <node_role>
    machine.openshift.io/cluster-api-machine-type: <node_role>
  name: <infrastructure_ID>-<node_role>
  namespace: openshift-machine-api
spec:
  replicas: <number_of_replicas>
  selector:
    matchLabels:
      machine.openshift.io/cluster-api-cluster: <infrastructure_ID>
      machine.openshift.io/cluster-api-machineset: <infrastructure_ID>-<node_role>
  template:
    metadata:
      labels:
        machine.openshift.io/cluster-api-cluster: <infrastructure_ID>
        machine.openshift.io/cluster-api-machine-role: <node_role>
        machine.openshift.io/cluster-api-machine-type: <node_role>
        machine.openshift.io/cluster-api-machineset: <infrastructure_ID>-<node_role>
    spec:
      providerSpec:
        value:
          apiVersion: openstackproviderconfig.openshift.io/v1alpha1
          cloudName: openstack
          cloudsSecret:
            name: openstack-cloud-credentials
            namespace: openshift-machine-api
          flavor: <nova_flavor>
          image: <glance_image_name_or_location>
          serverGroupID: aaaaaaaa-bbbb-cccc-dddd-eeeeeeeeeeee (1)
          kind: OpenstackProviderSpec
          networks:
          - filter: {}
            subnets:
            - filter:
                name: <subnet_name>
                tags: openshiftClusterID=<infrastructure_ID>
          securityGroups:
          - filter: {}
```

```
    name: <infrastructure_ID>-<node_role>
  serverMetadata:
    Name: <infrastructure_ID>-<node_role>
    openshiftClusterID: <infrastructure_ID>
  tags:
  - openshiftClusterID=<infrastructure_ID>
  trunk: true
  userDataSecret:
    name: <node_role>-user-data
  availabilityZone: <optional_openstack_availability_zone>
```

1 在此处添加服务器组的 UUID。

5. 可选：备份 **manifests/99_openshift-cluster-api_worker-machineset-0.yaml** 文件。创建集群时，安装程序会删除 **manifests/** 目录。

安装集群时，安装程序将使用您修改的 **MachineSet** 定义在 RHOSP 服务器组中创建计算机器。

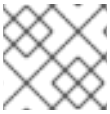
9.5.9. 生成 SSH 私钥并将其添加到代理中

如果要在集群上执行安装调试或灾难恢复，则必须为 **ssh-agent** 和安装程序提供 SSH 密钥。您可以使用此密钥访问公共集群中的 bootstrap 机器来排除安装问题。

注意

在生产环境中，您需要进行灾难恢复和调试。

您可以使用此密钥以 **core** 用户身份通过 SSH 连接到 master 节点。在部署集群时，此密钥会添加到 **core** 用户的 **~/.ssh/authorized_keys** 列表中。



注意

您必须使用一个本地密钥，而不要使用在特定平台上配置的密钥，如 AWS 密钥对。

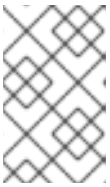
流程

1. 如果还没有为计算机上免密码身份验证而配置的 SSH 密钥，请创建一个。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ ssh-keygen -t ed25519 -N '' \
  -f <path>/<file_name> 1
```

1 指定新 SSH 密钥的路径和文件名，如 **~/.ssh/id_rsa**。如果您已有密钥对，请确保您的公钥位于 **~/.ssh** 目录中。

运行此命令会在指定的位置生成不需要密码的 SSH 密钥。



注意

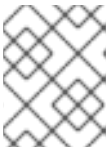
如果您计划在 **x86_64** 架构中安装使用 FIPS 验证的/Modules in Process 加密库的 OpenShift Container Platform 集群，不要创建使用 **ed25519** 算法的密钥。反之，创建一个使用 **rsa** 或 **ecdsa** 算法的密钥。

2. 作为后台任务启动 **ssh-agent** 进程：

```
$ eval "$(ssh-agent -s)"
```

输出示例

```
Agent pid 31874
```



注意

如果您的集群采用 FIPS 模式，则只使用 FIPS 兼容算法来生成 SSH 密钥。密钥必须是 RSA 或 ECDSA。

3. 将 SSH 私钥添加到 **ssh-agent**：

```
$ ssh-add <path>/<file_name> 1
```

输出示例

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

1 指定 SSH 私钥的路径和文件名，如 **~/.ssh/id_rsa**

后续步骤

- 在安装 OpenShift Container Platform 时，为安装程序提供 SSH 公钥。

9.5.10. 启用对环境的访问

在部署时，所有 OpenShift Container Platform 机器都是在 Red Hat OpenStack Platform (RHOSP) 租户网络中创建的。因此，大多数 RHOSP 部署中都无法直接访问它们。

您可以在安装过程中使用浮动 IP 地址（FIP）来配置 OpenShift Container Platform API 和应用程序访问。您也可以在没有配置 FIP 的情况下完成安装，但安装程序不会配置一种从外部访问 API 或应用程序的方法。

9.5.10.1. 启用通过浮动 IP 地址进行访问

创建浮动 IP（FIP）地址，用于从外部访问 OpenShift Container Platform API 和集群应用程序。

流程

1. 使用 Red Hat OpenStack Platform (RHOSP) CLI，创建 API FIP：

```
$ openstack floating ip create --description "API <cluster_name>.<base_domain>"
<external_network>
```

2. 使用 Red Hat OpenStack Platform (RHOSP) CLI，创建应用程序或 Ingress，FIP：

```
$ openstack floating ip create --description "Ingress <cluster_name>.<base_domain>"
<external_network>
```

3. 向用于 API 和 Ingress FIP 的 DNS 服务器添加符合这些模式的记录：

```
api.<cluster_name>.<base_domain>.  IN  A  <API_FIP>
*.apps.<cluster_name>.<base_domain>. IN  A <apps_FIP>
```

注意

如果您不控制 DNS 服务器，您可以通过将集群域名（如以下内容）添加到 **/etc/hosts** 文件中来访问集群：

- **<api_floating_ip> api.<cluster_name>.<base_domain>**
- **<application_floating_ip> grafana-openshift-monitoring.apps.<cluster_name>.<base_domain>**
- **<application_floating_ip> prometheus-k8s-openshift-monitoring.apps.<cluster_name>.<base_domain>**
- **<application_floating_ip> oauth-openshift.apps.<cluster_name>.<base_domain>**
- **<application_floating_ip> console-openshift-console.apps.<cluster_name>.<base_domain>**
- **application_floating_ip integrate-oauth-server-openshift-authentication.apps.<cluster_name>.<base_domain>**

/etc/hosts 文件中的集群域名授予对本地集群的 Web 控制台和监控界面的访问权限。您还可以使用 **kubectl** 或 **oc**。您可以使用指向 <application_floating_ip> 的额外条目来访问用户应用程序。此操作使 API 和应用程序可供您访问，不适合于生产部署，但允许对开发和测试进行安装。

4. 将 FIP 添加到 **install-config.yaml** 文件，将其作为以下参数的值：
   - **platform.openstack.ingressFloatingIP**
   - **platform.openstack.lbFloatingIP**

如果使用这些值，还必须在 **install-config.yaml** 文件中输入一个外部网络作为 **platform.openstack.externalNetwork** 参数的值。

提示

您可以通过分配浮动 IP 地址并更新防火墙配置，使 OpenShift Container Platform 资源在集群之外可用。

9.5.10.2. 完成没有浮动 IP 地址的安装

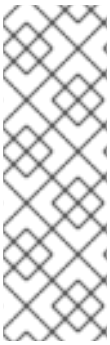
您可以在不提供浮动 IP 地址的情况下在 Red Hat OpenStack Platform（RHOSP）上安装 OpenShift Container Platform。

在 **install-config.yaml** 文件中，不要定义以下参数：

- **platform.openstack.ingressFloatingIP**
- **platform.openstack.lbFloatingIP**

如果您无法提供外部网络，也可以将 **platform.openstack.externalNetwork** 留空。如果没有为 **platform.openstack.externalNetwork** 提供值，则不会为您创建路由器。如果没有额外的操作，安装程序将无法从 Glance 检索镜像。您必须自行配置外部连接。

如果在因为缺少浮动 IP 地址或名称解析而无法访问集群 API 的系统中运行安装程序时，安装会失败。要防止安装失败，可以使用代理网络或者从与您的机器位于同一网络的系统中运行安装程序。



注意

您可以通过为 API 和 Ingress 端口创建 DNS 记录来启用名称解析。例如：

```
api.<cluster_name>.<base_domain>.  IN  A  <api_port_IP>
*.apps.<cluster_name>.<base_domain>. IN  A <ingress_port_IP>
```

如果您不控制 DNS 服务器，可以改为将记录添加到 **/etc/hosts** 文件中。此操作使 API 可供您自己访问，不适合于生产部署。这可用于进行开发和测试的安装。

9.5.11. 部署集群

您可以在兼容云平台中安装 OpenShift Container Platform。

重要

安装程序的 **create cluster** 命令只能在初始安装过程中运行一次。

先决条件

- 获取 OpenShift Container Platform 安装程序以及集群的 pull secret。

流程

1. 更改为包含安装程序的目录并初始化集群部署：

```
$ ./openshift-install create cluster --dir <installation_directory> \ 1
    --log-level=info 2
```

1. 对于 **<installation_directory>**，请指定自定义 **./install-config.yaml** 文件的位置。
2. 要查看不同的安装详情，请指定 **warn**、**debug** 或 **error**，而不要指定 **info**。

注意

如果您在主机上配置的云供应商帐户没有足够的权限来部署集群，安装过程将会停止，并且显示缺少的权限。

集群部署完成后，终端会显示访问集群的信息，包括指向其 Web 控制台的链接和 **kubeadmin** 用户的凭证。

输出示例

```
...
INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run 'export
KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'
INFO Access the OpenShift web-console here: https://console-openshift-
console.apps.mycluster.example.com
INFO Login to the console with user: "kubeadmin", and password: "4vYBz-Ee6gm-ymBZj-
Wt5AL"
INFO Time elapsed: 36m22s
```

注意

当安装成功时，集群访问和凭证信息还会输出到 **<installation_directory>/.openshift_install.log**。

重要

- 安装程序生成的 Ignition 配置文件包含在 24 小时后过期的证书，然后在过期时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外情况是，您需要手动批准待处理的 **node-bootstrapper** 证书签名请求（CSR）来恢复 kubelet 证书。如需更多信息，请参阅*从过期的 control plane 证书中恢复*的文档。
- 建议您在生成 12 小时后使用 Ignition 配置文件，因为集群安装后 24 小时证书从 16 小时轮转至 22 小时。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中运行证书更新时避免安装失败。

重要

您不得删除安装程序或安装程序所创建的文件。需要这两者才能删除集群。

9.5.12. 验证集群状态

您可以在安装过程中或安装后验证 OpenShift Container Platform 集群的状态：

流程

1. 在集群环境中，导出管理员的 kubeconfig 文件：

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

kubeconfig 文件包含关于集群的信息，供 CLI 用于将客户端连接到正确集群和 API 服务器。

2. 查看部署后创建的 control plane 和计算机器：

```
$ oc get nodes
```

3. 查看集群的版本：

-

```
$ oc get clusterversion
```

4. 查看 Operator 的状态：

```
$ oc get clusteroperator
```

5. 查看集群中的所有正在运行的 pod:

```
$ oc get pods -A
```

9.5.13. 使用 CLI 登录到集群

您可以通过导出集群 **kubeconfig** 文件，以默认系统用户身份登录集群。**kubeconfig** 文件包含关于集群的信息，供 CLI 用于将客户端连接到正确集群和 API 服务器。该文件特只适用于一个特定的集群，在 OpenShift Container Platform 安装过程中创建。

先决条件

- 已部署了 OpenShift Container Platform 集群。
- 已安装 **oc** CLI。

流程

1. 导出 **kubeadmin** 凭证：

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

2. 使用导出的配置，验证能否成功运行 **oc** 命令：

```
$ oc whoami
```

输出示例

```
system:admin
```

其他资源

- 如需有关访问和了解 OpenShift Container Platform Web 控制台的更多信息，请参阅访问 Web 控制台。

9.5.14. 禁用默认的 OperatorHub 源

在 OpenShift Container Platform 安装过程中，默认为 OperatorHub 配置由红帽和社区项目提供的源内容的 operator 目录。在受限网络环境中，必须以集群管理员身份禁用默认目录。

流程

- 通过在 **OperatorHub** 对象中添加 **disableAllDefaultSources: true** 来禁用默认目录的源：

```
$ oc patch OperatorHub cluster --type json \
    -p '[{"op": "add", "path": "/spec/disableAllDefaultSources", "value": true}]'
```

提示

或者，您可以使用 Web 控制台管理目录源。在 **Administration → Cluster Settings → Global Configuration → OperatorHub** 页面中，点 **Sources** 选项卡，其中可创建、删除、禁用和启用单独的源。

9.5.15. OpenShift Container Platform 的 Telemetry 访问

在 OpenShift Container Platform 4.6 中，默认运行的 Telemetry 服务提供有关集群健康状况和成功更新的指标，需要访问互联网。如果您的集群连接到互联网，Telemetry 会自动运行，而且集群会注册到 OpenShift Cluster Manager。

确认 OpenShift Cluster Manager 清单正确后，可以由 Telemetry 自动维护，也可以使用 OpenShift Cluster Manager 手动维护，使用订阅监控来跟踪帐户或多集群级别的 OpenShift Container Platform 订阅。

其他资源

- 有关 Telemetry 服务的更多信息，请参阅关于远程健康监控。

9.5.16. 后续步骤

- 自定义集群。
- 如果您用来安装集群的镜像 registry 具有一个可信任的 CA，通过配置额外的信任存储将其添加到集群中。
- 如果需要，您可以选择不使用远程健康报告。
- 为 Cluster Samples Operator 和 **must-gather** 工具配置镜像流。
- 了解如何在受限网络中使用 Operator Lifecycle Manager（OLM）。
- 如果您没有将 RHOSP 配置为使用浮动 IP 地址接受应用程序流量，使用浮动 IP 地址配置 RHOSP 访问。

9.6. 在 OPENSTACK 上卸载集群

您可以删除部署到 Red Hat OpenStack Platform (RHOSP) 的集群。

9.6.1. 删除使用安装程序置备的基础架构的集群

您可以从云中删除使用安装程序置备的基础架构的集群。

注意

卸载后，检查云供应商是否有没有被正确移除的资源，特别是 User Provisioned Infrastructure（UPI）集群。可能存在安装程序没有创建的资源，或者安装程序无法访问的资源。

先决条件

- 有部署集群时所用的安装程序副本。
- 有创建集群时安装程序所生成的文件。

流程

1. 在用来安装集群的计算机中包含安装程序的目录中，运行以下命令：

```
$ ./openshift-install destroy cluster \
--dir <installation_directory> --log-level info 1 2
```

1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

2 要查看不同的详情，请指定 **warn**、**debug** 或 **error**，而不要指定 **info**。

注意

您必须为集群指定包含集群定义文件的目录。安装程序需要此目录中的 **metadata.json** 文件来删除集群。

2. 可选：删除 **<installation_directory>** 目录和 OpenShift Container Platform 安装程序。

9.7. 从您自己的基础架构中卸载 RHOSP 上的集群

您可以删除在用户置备的基础架构上部署到 Red Hat OpenStack Platform（RHOSP）中的集群。

9.7.1. 下载 playbook 的依赖项

用于简化从用户置备的基础架构中移除过程的 Ansible playbook 需要几个 Python 模块。在您要进行这个操作的机器上添加模块的仓库，然后下载它们。

注意

这些说明假设您使用 Red Hat Enterprise Linux（RHEL）8。

先决条件

- Python 3 已安装在您的机器上。

流程

1. 在命令行中添加软件仓库：
   a. 使用 Red Hat Subscription Manager 注册：

```
$ sudo subscription-manager register # If not done already
```

   b. 获取最新的订阅数据：

```
$ sudo subscription-manager attach --pool=$YOUR_POOLID # If not done already
```

c. 禁用当前的软件仓库：

```
$ sudo subscription-manager repos --disable=* # If not done already
```

d. 添加所需的软件仓库：

```
$ sudo subscription-manager repos \
  --enable=rhel-8-for-x86_64-baseos-rpms \
  --enable=openstack-16-tools-for-rhel-8-x86_64-rpms \
  --enable=ansible-2.9-for-rhel-8-x86_64-rpms \
  --enable=rhel-8-for-x86_64-appstream-rpms
```

2. 安装模块：

```
$ sudo yum install python3-openstackclient ansible python3-openstacksdk
```

3. 确保 **python** 命令指向 **python3**:

```
$ sudo alternatives --set python /usr/bin/python3
```

9.7.2. 从使用您自己的基础架构的 RHOSP 中删除集群

您可以在使用您自己的基础架构的 Red Hat OpenStack Platform（RHOSP）上删除 OpenShift Container Platform 集群。要快速完成移除过程，请运行多个 Ansible playbook。

先决条件

- Python 3 已安装在您的机器上。
- 您下载了"下载 playbook 依赖项"中的模块。
- 您有用于安装集群的 playbook。
- 已修改前缀为 **down-** 的 playbook，以反映您对相应安装 playbook 所做的任何更改。例如，对 **bootstrap.yaml** 文件的改变会反映在 **down-bootstrap.yaml** 文件中。
- 所有 playbook 都位于一个通用目录中。

流程

1. 在命令行中运行您下载的 playbook：

```
$ ansible-playbook -i inventory.yaml  \
 down-bootstrap.yaml      \
 down-control-plane.yaml  \
 down-compute-nodes.yaml  \
 down-load-balancers.yaml \
 down-network.yaml        \
 down-security-groups.yaml
```

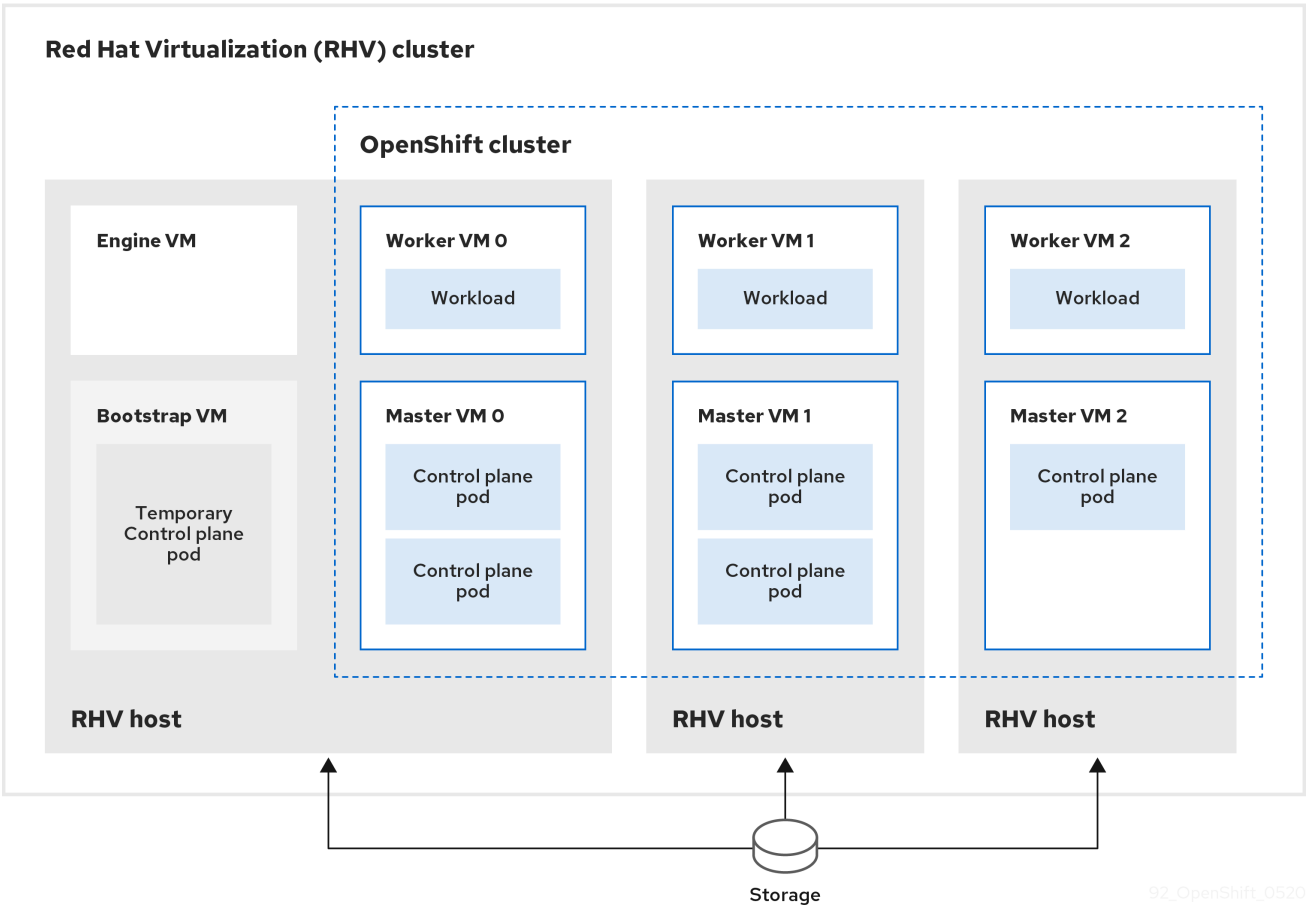
2. 删除您为 OpenShift Container Platform 安装所做的任何 DNS 记录更改。

OpenShift Container Platform 被从您的基础架构中删除。

第 10 章 在 RHV 上安装

10.1. 在 RHV 上快速安装集群

您可以快速地在 Red Hat Virtualization（RHV）集群中安装默认、非自定义的 OpenShift Container Platform 集群，如下图所示。



安装程序使用安装程序置备的基础架构自动创建和部署集群。

要安装默认集群，请准备环境，运行安装程序并根据提示提供相应信息。然后，安装程序会创建 OpenShift Container Platform 集群。

有关安装默认集群的其他方法，请参阅 使用自定义安装集群。

注意

这个安装程序只适用于 Linux 和 macOS。

10.1.1. 先决条件

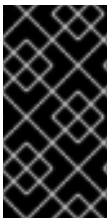
- 查看有关 OpenShift Container Platform 安装和更新 流程的详细信息。
- 在 Red Hat Virtualization(RHV)上的 OpenShift Container Platform Support Matrix 中支持的版本组合。
- 如果使用防火墙，则必须将其配置为允许集群需要访问的站点。

10.1.2. OpenShift Container Platform 的互联网访问

在 OpenShift Container Platform 4.6 中，您需要访问互联网来安装集群。

您必须具有以下互联网访问权限：

- 访问 OpenShift Cluster Manager 以下载安装程序并执行订阅管理。如果集群可以访问互联网，并且没有禁用 Telemetry，该服务会自动授权您的集群。
- 访问 Quay.io，以获取安装集群所需的软件包。
- 获取执行集群更新所需的软件包。



重要

如果您的集群无法直接访问互联网，则可以在置备的某些类基础架构上执行受限网络安装。在此过程中，您要下载所需的内容，并使用它在镜像 registry（mirror registry） 中填充安装集群并生成安装程序所需的软件包。对于某些安装类型，集群要安装到的环境不需要访问互联网。在更新集群之前，要更新 registry 镜像系统中的内容。

10.1.3. RHV 环境的要求

要安装并运行 OpenShift Container Platform 集群，RHV 环境必须满足以下要求。

不满足这些要求会导致安装或进程失败。另外，无法满足这些要求可能会导致 OpenShift Container Platform 集群在安装后几天或几星期后失败。

对 CPU、内存和存储资源的以下要求是基于**默认值**乘以安装程序创建的默认虚拟机数。除了 RHV 环境用于非 OpenShift Container Platform 操作的资源外，**这些资源还必须可用**。

默认情况下，安装程序会在安装过程中创建七台虚拟机。首先，它会创建一个 bootstrap 虚拟机来提供临时服务和 control plane，同时创建 OpenShift Container Platform 集群的其余部分。当安装程序完成集群创建时，删除 bootstrap 机器可释放其资源。

如果在 RHV 环境中增加虚拟机数量，则需要相应地增加资源。

要求

- RHV 环境有一个数据中心，其状态是 **Up**。
- RHV 数据中心包含一个 RHV 集群。
- RHV 集群具有专门用于 OpenShift Container Platform 集群的以下资源：
  - 最小 28 个 vCPU：在安装过程中创建的七个虚拟机，每个都需要 4 个。
  - 112 GiB RAM 或更多，包括：
    - 16 GiB 或更多用于提供临时 control plane 功能的 bootstrap 机器。
    - 每个提供 control plane 功能的三台 control plane 机器都需要16 GiB 或更多。
    - 每个用来运行应用程序负载的 compute 机器都需要 16 GiB 或更多。
- RHV 存储域必须满足 etcd 后端性能要求。
- 在生产环境中，每个虚拟机必须具有 120 GiB 或更多存储。因此，存储域必须为默认的

OpenShift Container Platform 集群提供 840 GiB 或更多存储。在资源有限制或非生产环境中，每个虚拟机必须具有 32 GiB 或更多存储，因此对于默认的 OpenShift Container Platform 集群，存储域必须具有 230 GiB 或更多存储 。

- 要在安装和更新过程中从红帽生态系统目录下载镜像，RHV 集群必须可以访问互联网。Telemetry 服务还需要互联网连接来简化订阅和权利的过程。
- RHV 集群需要有一个虚拟网络，可访问 RHV Manager 上的 REST API。确保在这个网络中启用了 DHCP，因为安装程序创建的虚拟机会使用 DHCP 来获取他们的 IP 地址。
- 具有以下最少权限的用户帐户和组，用于在目标 RHV 集群上安装和管理 OpenShift Container Platform 集群：
  - **DiskOperator**
  - **DiskCreator**
  - **UserTemplateBasedVm**
  - **TemplateOwner**
  - **TemplateCreator**
  - 目标集群的**ClusterAdmin**

警告

使用最少权限：在安装过程中，避免使用带有 RHV **SuperUser** 权限的管理员帐户。安装程序会将您提供的凭据保存到一个临时的 **ovirt-config.yaml** 文件中，这个凭证有被受到破坏的可能。

10.1.4. 验证 RHV 环境的要求

验证 RHV 环境是否满足安装和运行 OpenShift Container Platform 集群的要求。不满足这些要求会导致问题。

重要

这些要求基于安装程序用来创建 control plane 和计算机器的默认资源。这些资源包括 vCPU、内存和存储。如果更改这些资源或增加 OpenShift Container Platform 机器的数量，请相应调整这些要求。

流程

1. 检查 RHV 版本。
   a. 在 RHV 管理门户中，点右上角的 ? 帮助图表，选 **About**。
   b. 在打开的窗口中，记录下 **RHV 软件版本**。

c. 确认 OpenShift Container Platform 版本 4.6 和您记录的 RHV 版本组合是被支持的。 RHV 上支持的 OpenShift Container Platform。

2. 检查数据中心、集群和存储。

a. 在 RHV 管理门户中，点 **Compute → Data Centers**。

b. 确认可以访问您要安装 OpenShift Container Platform 的数据中心。

c. 点击该数据中心的名称。

d. 在数据中心详情中，**存储** 标签中确认您要安装 OpenShift Container Platform 的存储域是 **Active**。

e. 记录下**域名**以供以后使用。

f. 确认 **Free Space** 至少为 230 GiB。

g. 确认存储域满足 etcd 后端性能要求，可以使用 fio 性能基准工具来评测。

h. 在数据中心详情中点击 **Clusters** 选项卡。

i. 找到您要安装 OpenShift Container Platform 的 RHV 集群。记录集群名称，以供稍后使用。

3. 检查 RHV 主机资源。

a. 在 RHV 管理门户中，点 **Compute > Clusters**。

b. 点击要安装 OpenShift Container Platform 的集群。

c. 在集群详情中点击 **Hosts** 标签页。

d. 检查主机，确认这些主机有至少 28 个 **逻辑 CPU 内核**，专门用于 OpenShift Container Platform 集群。

e. 记录**逻辑 CPU 内核**数以供稍后使用。

f. 请确认这些 CPU 内核被正确分配，在安装过程中创建的七台虚拟机中的每一台都可以有四个内核。

g. 确认主机总共有 112 GiB 的**Max free Memory for scheduling new virtual machines**，以满足以下每个 OpenShift Container Platform 机器的要求：

- bootstrap 机器需要 16 GiB
- 三个 control plane 机器每个机器都需要 16 GiB
- 三个计算机器每个机器都需要 16 GiB

h. 记录下 **Max free Memory for scheduling new virtual machine**的值以便稍后使用。

4. 验证安装 OpenShift Container Platform 的虚拟网络能否访问 RHV Manager 的 REST API。在这个网络的虚拟机上，使用 curl 来访问 RHV Manager 的 REST API:

```
$ curl -k -u <username>@<profile>:<password> \ 1
https://<engine-fqdn>/ovirt-engine/api 2
```

1. 对于 **<username>**，指定具有在 RHV 上创建和管理 OpenShift Container Platform 集群的 RHV 帐户的用户名。对于 **<profile>**，请指定登录配置集，您可以登陆到 RHV 管理门户查看 **Profile** 下拉列表。对于 **<password>**，指定该用户的密码。

2. 对于 **<engine-fqdn>**，请指定 RHV 环境的完全限定域名。

例如：

```
$ curl -k -u ocpadmin@internal:pw123 \
https://rhv-env.virtlab.example.com/ovirt-engine/api
```

10.1.5. 在 RHV 中准备网络环境

为 OpenShift Container Platform 集群配置两个静态 IP 地址，并使用这些地址创建 DNS 条目。

流程

1. 保留两个静态 IP 地址

   a. 在您要安装 OpenShift Container Platform 的网络上，标识 DHCP 租期池之外的两个静态 IP 地址。

   b. 连接到此网络中的主机，并确认每个 IP 地址都没有被使用。例如，使用地址解析协议（ARP）检查 IP 地址是否有条目：

   ```
   $ arp 10.35.1.19
   ```

   输出示例

   ```
   10.35.1.19 (10.35.1.19) -- no entry
   ```

   c. 为您的网络环境保留两个静态 IP 地址。

   d. 记录这些 IP 地址以备将来参考。

2. 为 OpenShift Container Platform REST API 创建 DNS 条目，并使用以下格式应用域名：

```
api.<cluster-name>.<base-domain>   <ip-address> 1
*.apps.<cluster-name>.<base-domain>   <ip-address> 2
```

1. 对于 **<cluster-name>**、**<base-domain>**和 **<ip-address>**，请指定 OpenShift Container Platform API 的集群名称、基域和静态 IP 地址。

2. 指定 Ingress 和负载均衡器的 OpenShift Container Platform 应用程序的集群名称、基域和静态 IP 地址。

例如：

```
api.my-cluster.virtlab.example.com 10.35.1.19
*.apps.my-cluster.virtlab.example.com 10.35.1.20
```

10.1.6. 为 RHV 设置 CA 证书

从 Red Hat Virtualization (RHV) Manager 下载 CA 证书，并在安装机器中进行设置。

您可以使用 RHV Manager 的网页或使用 **curl** 命令下载该证书。

之后，您向安装程序提供证书。

流程

1. 使用这两个方法之一下载 CA 证书：
   - 进入 Manager 的网页 **https://<engine-fqdn>/ovirt-engine/**。然后在 下载 中点击 CA 证书 链接。
   - 运行以下命令：

     ```
     $ curl -k 'https://<engine-fqdn>/ovirt-engine/services/pki-resource?resource=ca-certificate&format=X509-PEM-CA' -o /tmp/ca.pem
     ```
     (1)

     (1) 对于 **<engine-fqdn>**，请指定 RHV Manager 的全限定域名，如 **rhv-env.virtlab.example.com**。

2. 配置 CA 文件，为 Manager 授予无根用户访问权限。将 CA 文件权限设置为 **0644** （symbolic 值： **-rw-r—r--**）：

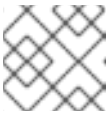
   ```
   $ sudo chmod 0644 /tmp/ca.pem
   ```

3. 对于 Linux，将 CA 证书复制到服务器证书目录中。使用 **-p** 保留权限：

   ```
   $ sudo cp -p /tmp/ca.pem /etc/pki/ca-trust/source/anchors/ca.pem
   ```

4. 将证书添加到您操作系统的证书管理器：
   - 对于 macOS，请双击这个证书文件，并使用 **Keychain Access** 程序将该文件添加到 **System** 密钥链中。
   - 对于 Linux，更新 CA 信任：

     ```
     $ sudo update-ca-trust
     ```

     注意

     如果使用您自己的证书认证机构，请确定系统信任它。

其他资源

- 如需了解更多相关信息，请参阅 RHV 文档中的 身份验证及安全性。

10.1.7. 生成 SSH 私钥并将其添加到代理中

如果要在集群上执行安装调试或灾难恢复，则必须为 **ssh-agent** 和安装程序提供 SSH 密钥。您可以使用此密钥访问公共集群中的 bootstrap 机器来排除安装问题。

注意

在生产环境中，您需要进行灾难恢复和调试。

您可以使用此密钥以 **core** 用户身份通过 SSH 连接到 master 节点。在部署集群时，此密钥会添加到 **core** 用户的 **~/.ssh/authorized_keys** 列表中。

流程

1. 如果还没有为计算机上免密码身份验证而配置的 SSH 密钥，请创建一个。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ ssh-keygen -t ed25519 -N '' \
  -f <path>/<file_name> 1
```

1 指定新 SSH 密钥的路径和文件名，如 **~/.ssh/id_rsa**。如果您已有密钥对，请确保您的公钥位于 **~/.ssh** 目录中。

运行此命令会在指定的位置生成不需要密码的 SSH 密钥。

注意

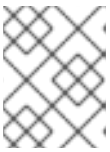
如果您计划在 **x86_64** 架构中安装使用 FIPS 验证的/Modules in Process 加密库的 OpenShift Container Platform 集群，不要创建使用 **ed25519** 算法的密钥。反之，创建一个使用 **rsa** 或 **ecdsa** 算法的密钥。

2. 作为后台任务启动 **ssh-agent** 进程：

```
$ eval "$(ssh-agent -s)"
```

输出示例

```
Agent pid 31874
```



注意

如果您的集群采用 FIPS 模式，则只使用 FIPS 兼容算法来生成 SSH 密钥。密钥必须是 RSA 或 ECDSA。

3. 将 SSH 私钥添加到 **ssh-agent**：

```
$ ssh-add <path>/<file_name> 1
```

输出示例

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

1 指定 SSH 私钥的路径和文件名，如 **~/.ssh/id_rsa**

后续步骤

- 在安装 OpenShift Container Platform 时，为安装程序提供 SSH 公钥。

10.1.8. 获取安装程序

在安装 OpenShift Container Platform 之前，将安装文件下载到本地计算机上。

先决条件

- 运行 Linux 或 macOS 的计算机，本地磁盘空间为 500 MB

流程

1. 访问 OpenShift Cluster Manager 站点的 Infrastructure Provider 页面。如果您有红帽帐号，请使用自己的凭证登录。如果没有，请创建一个帐户。
2. 选择您的基础架构供应商。
3. 进入适用于您的安装类型的页面，下载您的操作系统的安装程序，并将文件放在要保存安装配置文件的目录中。。

重要

安装程序会在用来安装集群的计算机上创建若干文件。在完成集群安装后，您必须保留安装程序和安装程序所创建的文件。这两个文件都需要删除集群。

重要

删除安装程序创建的文件不会删除您的集群，即使集群在安装过程中失败也是如此。要删除集群，为特定云供应商完成 OpenShift Container Platform 卸载流程。

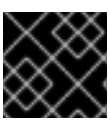
4. 提取安装程序。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ tar xvf openshift-install-linux.tar.gz
```

5. 从 Red Hat OpenShift Cluster Manager 下载安装 pull secret 。通过此 pull secret，您可以进行所含授权机构提供的服务的身份验证，这些服务包括为 OpenShift Container Platform 组件提供容器镜像的 Quay.io。

10.1.9. 部署集群

您可以在兼容云平台中安装 OpenShift Container Platform。



重要

安装程序的 **create cluster** 命令只能在初始安装过程中运行一次。

先决条件

- 从运行安装程序的机器中打开到 Manager 的 **ovirt-imageio** 端口。默认情况下，端口为 **54322**。
- 获取 OpenShift Container Platform 安装程序以及集群的 pull secret。

流程

1. 更改为包含安装程序的目录并初始化集群部署：

```
$ ./openshift-install create cluster --dir <installation_directory> \ 1
    --log-level=info 2
```

1 对于 **<installation_directory>**，请指定用于保存安装程序所创建的文件的目录名称。

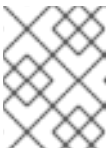
2 要查看不同的安装详情，请指定 **warn**、**debug** 或 **error**，而不要指定 **info**。

重要

指定一个空目录。一些安装信息，如 bootstrap X.509 证书，有较短的过期间隔，因此不要重复使用安装目录。如果要重复使用另一个集群安装中的个别文件，可以将其复制到您的目录中。但是，一些安装数据的文件名可能会在发行版本之间有所改变。从 OpenShift Container Platform 老版本中复制安装文件时要格外小心。

根据安装程序提示输入相关的值。

a. 可选： **SSH Public Key**，请选择无密码公钥，如 **~/.ssh/id_rsa.pub**。这个密钥被用来验证与新的 OpenShift Container Platform 集群的连接。



注意

如果您要在生产环境中执行安装调试或灾难恢复，请指定 **ssh-agent** 进程需要使用的 SSH 密钥。

b. 对于 **Platform**，选择 **ovirt**。

c. 对于 **Engine FQDN[:PORT]**，请输入 RHV 环境的完全限定域名（FQDN）。
例如：

```
rhv-env.virtlab.example.com:443
```

d. 安装程序会自动生成 CA 证书。对于 **Would you like to use the above certificate to connect to the Manager?**，输入 **y** 或 **N**。如果回答 **N**，则必须在不安全的模式安装 OpenShift Container Platform。

e. 对于 **Engine username**，请使用以下格式输入 RHV 管理员的用户名和配置集：

```
<username>@<profile> 1
```

1 对于 **<username>**，请指定 RHV 管理员的用户名。对于 **<profile>**，请指定登录配置集，您可以登陆到 RHV 管理门户查看 **Profile** 下拉列表。例如： **admin@internal**。

f. 对于 **Engine 密码**，请输入 RHV 管理密码。

g. 对于 **Cluster**，请选择用于安装 OpenShift Container Platform 的 RHV 集群。

h. 对于 **Storage domain**，请选择安装 OpenShift Container Platform 的存储域。

i. 对于 **Network**，选择可访问 RHV Manager REST API 的虚拟网络。

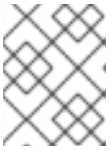
j. 对于 **Internal API Virtual IP**，请为集群的 REST API 输入您设置的静态 IP 地址。

k. 对于 **Ingress virtual IP**，请为通配符应用程序域输入您保留的静态 IP 地址。

l. 对于 **Base Domain**，请输入 OpenShift Container Platform 集群的基域。如果这个群集暴露于外部世界，这必须是 DNS 基础结构可识别的有效域。例如： 输入 **virtlab.example.com**

m. 对于 **Cluster Name**，请输入集群名称。例如： **my-cluster**。使用您为 OpenShift Container Platform REST API 创建的外部注册/可解析 DNS 条目的集群名称，以及应用域名。安装程序也将此名称提供给 RHV 环境中的集群。

n. 对于 **Pull Secret**，请从之前下载并粘贴的 **pull-secret.txt** 文件中复制 pull secret。您还可以从 Red Hat OpenShift Cluster Manager 获取同一 pull secret 的副本。



注意

如果您在主机上配置的云供应商帐户没有足够的权限来部署集群，安装过程将会停止，并且显示缺少的权限。

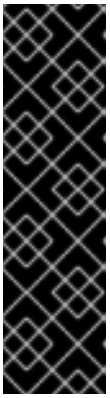
集群部署完成后，终端会显示访问集群的信息，包括指向其 Web 控制台的链接和 **kubeadmin** 用户的凭证。

输出示例

```
...
INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run 'export
KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'
INFO Access the OpenShift web-console here: https://console-openshift-
console.apps.mycluster.example.com
INFO Login to the console with user: "kubeadmin", and password: "4vYBz-Ee6gm-ymBZj-
Wt5AL"
INFO Time elapsed: 36m22s
```

注意

当安装成功时，集群访问和凭证信息还会输出到 **<installation_directory>/.openshift_install.log**。



重要

- 安装程序生成的 Ignition 配置文件包含在 24 小时后过期的证书，然后在过期时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外情况是，您需要手动批准待处理的 **node-bootstrapper** 证书签名请求（CSR）来恢复 kubelet 证书。如需更多信息，请参阅*从过期的 control plane 证书中恢复*的文档。

- 建议您在生成 12 小时后使用 Ignition 配置文件，因为集群安装后 24 小时证书从 16 小时轮转至 22 小时。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中运行证书更新时避免安装失败。

重要

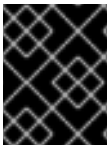
您不得删除安装程序或安装程序所创建的文件。需要这两者才能删除集群。

重要

您已完成了安装集群所需的步骤。余下的步骤演示了如何验证集群并对安装进行故障排除。

10.1.10. 通过下载二进制文件安装 OpenShift CLI

您需要安装 CLI（**oc**） 来使用命令行界面与 OpenShift Container Platform 进行交互。您可在 Linux 、Windows 或 macOS 上安装 **oc**。



重要

如果安装了旧版本的 **oc**，则无法使用 OpenShift Container Platform 4.6 中的所有命令。下载并安装新版本的 **oc**。

10.1.10.1. 在 Linux 上安装 OpenShift CLI

您可以按照以下流程在 Linux 上安装 OpenShift CLI（**oc**）二进制文件。

流程

1. 进入到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 在 **Version** 下拉菜单中选择相应的版本。
3. 单击 **OpenShift v4.6 Linux 客户端** 条目旁边的 **Download Now**，再保存文件。
4. 解包存档：

```
$ tar xvzf <file>
```

5. 把 **oc** 二进制代码放到 **PATH** 中的目录中。
   执行以下命令可以查看当前的 **PATH** 设置：

```
$ echo $PATH
```

安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
$ oc <command>
```

10.1.10.2. 在 Windows 上安装 OpenShift CLI

您可以按照以下流程在 Windows 上安装 OpenShift CLI（**oc**）二进制代码。

流程

1. 进入到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 在 **Version** 下拉菜单中选择相应的版本。

3. 单击 **OpenShift v4.6 Windows 客户端**条目旁边的 **Download Now**，再保存文件。
4. 使用 ZIP 程序解压存档。
5. 把 **oc** 二进制代码放到 **PATH** 中的目录中。
   要查看您的 **PATH**，请打开命令提示窗口并执行以下命令：

```
C:\> path
```

安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
C:\> oc <command>
```

10.1.10.3. 在 macOS 上安装 OpenShift CLI

您可以按照以下流程在 macOS 上安装 OpenShift CLI（**oc**）二进制代码。

流程

1. 进入到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 在 **Version** 下拉菜单中选择相应的版本。
3. 单击 **OpenShift v4.6 MacOSX 客户端**条目旁边的 **Download Now**，再保存文件。
4. 解包和解压存档。
5. 将 **oc** 二进制文件移到 PATH 的目录中。
   要查看您的 **PATH**，打开一个终端窗口并执行以下命令：

```
$ echo $PATH
```

安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
$ oc <command>
```

如需更多信息，请参阅 OpenShift CLI 入门。

10.1.11. 使用 CLI 登录到集群

您可以通过导出集群 **kubeconfig** 文件，以默认系统用户身份登录集群。**kubeconfig** 文件包含关于集群的信息，供 CLI 用于将客户端连接到正确集群和 API 服务器。该文件特只适用于一个特定的集群，在 OpenShift Container Platform 安装过程中创建。

先决条件

- 已部署了 OpenShift Container Platform 集群。
- 已安装 **oc** CLI。

流程

1. 导出 **kubeadmin** 凭证：

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

2. 使用导出的配置，验证能否成功运行 **oc** 命令：

```
$ oc whoami
```

输出示例

```
system:admin
```

其他资源

- 如需有关访问和了解 OpenShift Container Platform Web 控制台的更多信息，请参阅访问 Web 控制台。

10.1.12. 验证集群状态

您可以在安装过程中或安装后验证 OpenShift Container Platform 集群的状态：

流程

1. 在集群环境中，导出管理员的 kubeconfig 文件：

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

kubeconfig 文件包含关于集群的信息，供 CLI 用于将客户端连接到正确集群和 API 服务器。

2. 查看部署后创建的 control plane 和计算机器：

```
$ oc get nodes
```

3. 查看集群的版本：

```
$ oc get clusterversion
```

4. 查看 Operator 的状态：

```
$ oc get clusteroperator
```

5. 查看集群中的所有正在运行的 pod:

```
$ oc get pods -A
```

故障排除

如果安装失败，安装程序会超时并显示出错信息。如需了解更多相关信息，请参阅故障排除安装问题。

10.1.13. 访问 RHV 上的 OpenShift Container Platform Web 控制台。

OpenShift Container Platform 集群初始化后，您可以登录到 OpenShift Container Platform Web 控制台。

流程

1. 可选：在 Red Hat Virtualization (RHV) 管理门户中，打开 **Compute → Cluster**。
2. 验证安装程序是否创建了虚拟机。
3. 返回到安装程序正在运行的命令行。当安装程序完成后，它会显示登录到 OpenShift Container Platform Web 控制台的用户名和临时密码。
4. 在浏览器中，打开 OpenShift Container Platform web 控制台的 URL。URL 使用以下格式：

```
console-openshift-console.apps.<clustername>.<basedomain> 1
```

1 对于 **<clustername>.<baseDomain>**，请指定集群名称和基域。

例如：

```
console-openshift-console.apps.my-cluster.virtlab.example.com
```

10.1.14. OpenShift Container Platform 的 Telemetry 访问

在 OpenShift Container Platform 4.6 中，默认运行的 Telemetry 服务提供有关集群健康状况和成功更新的指标，需要访问互联网。如果您的集群连接到互联网，Telemetry 会自动运行，而且集群会注册到 OpenShift Cluster Manager。

确认 OpenShift Cluster Manager 清单正确后，可以由 Telemetry 自动维护，也可以使用 OpenShift Cluster Manager 手动维护，使用订阅监控来跟踪帐户或多集群级别的 OpenShift Container Platform 订阅。

其他资源

- 有关 Telemetry 服务的更多信息，请参阅关于远程健康监控。

10.1.15. 在 Red Hat Virtualization (RHV) 上安装时的常见问题

以下是您可能会遇到的一些常见问题，以及推荐的原因和解决方案。

10.1.15.1. CPU 负载增加和节点进入非就绪状态

- **症状**: CPU 负载显著增加，节点开始处于 **Not Ready** 状态。
- **原因**: 存储域延迟可能太大，特别是针对 control plane 节点（也称为 master 节点）。
- **解决方案**:
  通过重启 kubelet 服务使节点再次就绪：

```
$ systemctl restart kubelet
```

检查 OpenShift Container Platform 指标服务，该服务可自动收集并报告一些重要数据，如 etcd 磁盘同步持续时间。如果集群是可操作的，使用这个数据来帮助确定这个问题是否是因为存储延迟或吞吐量造成的。如果是这样，请考虑使用一个较低延迟和更高吞吐量的存储资源。

要获得原始指标，请以 kubeadmin 或具有 cluster-admin 特权的用户身份输入以下命令：

```
$ oc get --insecure-skip-tls-verify --server=https://localhost:<port> --raw=/metrics
```

如需了解更多相关信息，请参阅 使用 OpenShift 4.x 调试应用程序端点 。

10.1.15.2. 连接到 OpenShift Container Platform 集群 API 存在问题

- **症状**: 安装程序完成，但无法使用 OpenShift Container Platform 集群 API。在 bootstrap 过程完成后，bootstrap 虚拟机仍处于在线状态。当您输入以下命令时，回复会超时。

  ```
  $ oc login -u kubeadmin -p *** <apiurl>
  ```

- **原因**: 安装程序没有删除 bootstrap VM，因此没有释放集群的 API IP 地址。

- **解决方法**：使用 **wait-for** 子命令，在 bootstrap 过程完成后获得通知：

  ```
  $ ./openshift-install wait-for bootstrap-complete
  ```

  当 bootstrap 过程完成后，删除 bootstrap 虚拟机：

  ```
  $ ./openshift-install destroy bootstrap
  ```

10.1.16. 安装后的任务

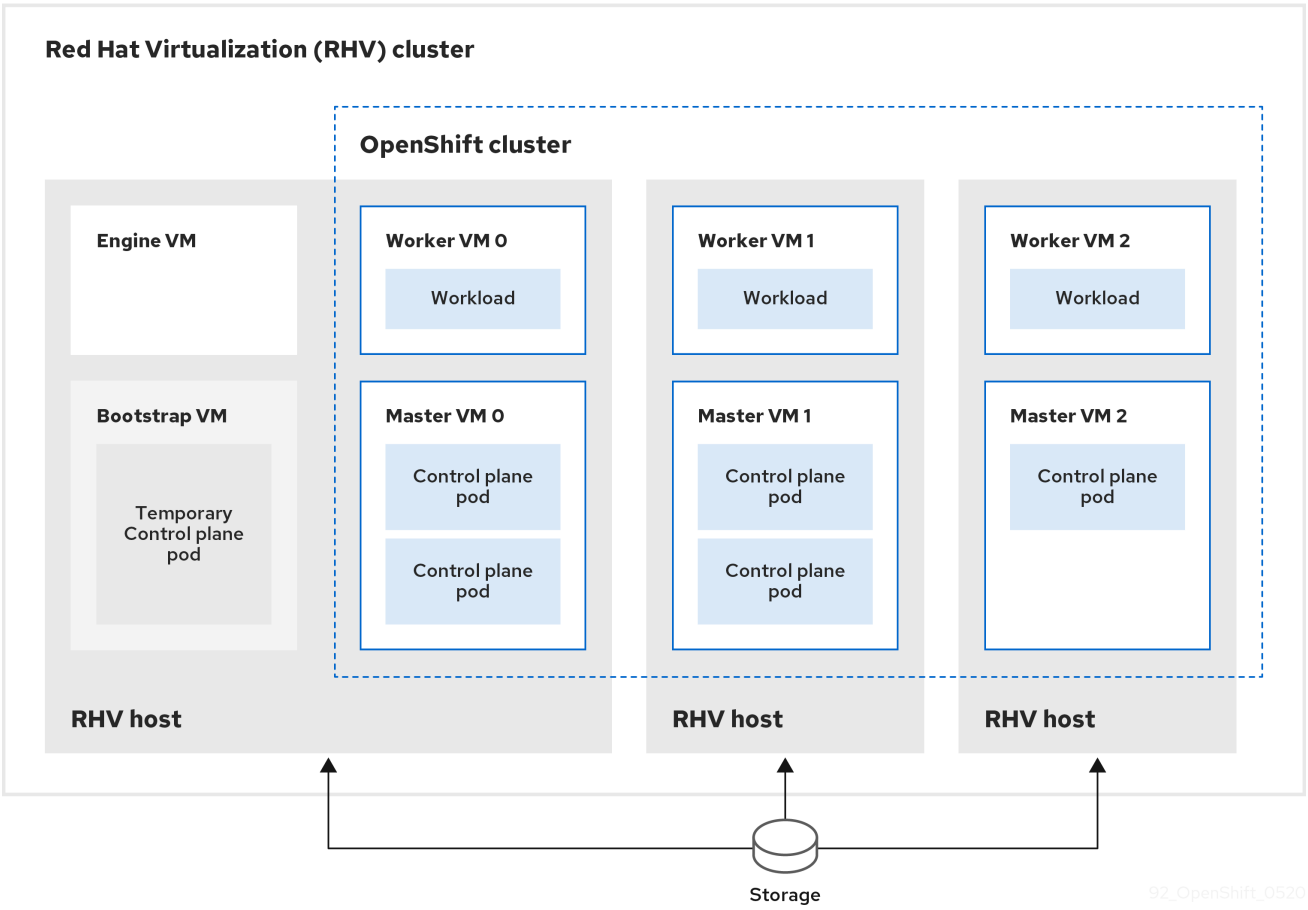
在 OpenShift Container Platform 集群初始化后，您可以执行以下任务。

- 可选： 在部署后，使用 OpenShift Container Platform 中的 Machine Config Operator (MCO) 添加或替换 SSH 密钥。

- 可选：删除 **kubeadmin** 用户。使用身份验证提供程序创建具有 cluster-admin 权限的用户。

10.2. 使用自定义在 RHV 上安装集群

您可以在 Red Hat Virtualization（RHV）上自定义并安装 OpenShift Container Platform 集群，如下图所示类似。



安装程序使用安装程序置备的基础架构自动创建和部署集群。

要安装自定义集群，请准备环境并执行以下步骤：

1. 通过运行安装程序并根据提示提供信息来创建安装配置文件 **install-config.yaml**。
2. 检查并修改 **install-config.yaml** 文件中的参数。
3. 生成 **install-config.yaml** 文件的一个工作副本。
4. 在运行安装程序时，使用 **install-config.yaml** 文件的副本。

然后，安装程序会创建 OpenShift Container Platform 集群。

有关安装自定义集群的其他方法，请参阅安装默认集群。

注意

这个安装程序只适用于 Linux 和 macOS。

10.2.1. 先决条件

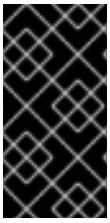
- 查看有关 OpenShift Container Platform 安装和更新流程的详细信息。
- 在 Red Hat Virtualization(RHV)上的 OpenShift Container Platform Support Matrix 中支持的版本组合。
- 如果使用防火墙，则必须将其配置为允许集群需要访问的站点。

10.2.2. OpenShift Container Platform 的互联网访问

在 OpenShift Container Platform 4.6 中，您需要访问互联网来安装集群。

您必须具有以下互联网访问权限：

- 访问 OpenShift Cluster Manager 以下载安装程序并执行订阅管理。如果集群可以访问互联网，并且没有禁用 Telemetry，该服务会自动授权您的集群。
- 访问 Quay.io，以获取安装集群所需的软件包。
- 获取执行集群更新所需的软件包。



重要

如果您的集群无法直接访问互联网，则可以在置备的某些类基础架构上执行受限网络安装。在此过程中，您要下载所需的内容，并使用它在镜像 registry（mirror registry） 中填充安装集群并生成安装程序所需的软件包。对于某些安装类型，集群要安装到的环境不需要访问互联网。在更新集群之前，要更新 registry 镜像系统中的内容。

10.2.3. RHV 环境的要求

要安装并运行 OpenShift Container Platform 集群，RHV 环境必须满足以下要求。

不满足这些要求会导致安装或进程失败。另外，无法满足这些要求可能会导致 OpenShift Container Platform 集群在安装后几天或几星期后失败。

对 CPU、内存和存储资源的以下要求是基于**默认值**乘以安装程序创建的默认虚拟机数。除了 RHV 环境用于非 OpenShift Container Platform 操作的资源外，**这些资源还必须可用**。

默认情况下，安装程序会在安装过程中创建七台虚拟机。首先，它会创建一个 bootstrap 虚拟机来提供临时服务和 control plane，同时创建 OpenShift Container Platform 集群的其余部分。当安装程序完成集群创建时，删除 bootstrap 机器可释放其资源。

如果在 RHV 环境中增加虚拟机数量，则需要相应地增加资源。

要求

- RHV 环境有一个数据中心，其状态是 **Up**。
- RHV 数据中心包含一个 RHV 集群。
- RHV 集群具有专门用于 OpenShift Container Platform 集群的以下资源：
  - 最小 28 个 vCPU：在安装过程中创建的七个虚拟机，每个都需要 4 个。
  - 112 GiB RAM 或更多，包括：
    - 16 GiB 或更多用于提供临时 control plane 功能的 bootstrap 机器。
    - 每个提供 control plane 功能的三台 control plane 机器都需要16 GiB 或更多。
    - 每个用来运行应用程序负载的 compute 机器都需要 16 GiB 或更多。
- RHV 存储域必须满足 etcd 后端性能要求。
- 在生产环境中，每个虚拟机必须具有 120 GiB 或更多存储。因此，存储域必须为默认的

OpenShift Container Platform 集群提供 840 GiB 或更多存储。在资源有限制或非生产环境中，每个虚拟机必须具有 32 GiB 或更多存储，因此对于默认的 OpenShift Container Platform 集群，存储域必须具有 230 GiB 或更多存储 。

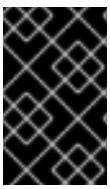
- 要在安装和更新过程中从红帽生态系统目录下载镜像，RHV 集群必须可以访问互联网。Telemetry 服务还需要互联网连接来简化订阅和权利的过程。
- RHV 集群需要有一个虚拟网络，可访问 RHV Manager 上的 REST API。确保在这个网络中启用了 DHCP，因为安装程序创建的虚拟机会使用 DHCP 来获取他们的 IP 地址。
- 具有以下最少权限的用户帐户和组，用于在目标 RHV 集群上安装和管理 OpenShift Container Platform 集群：
  - **DiskOperator**
  - **DiskCreator**
  - **UserTemplateBasedVm**
  - **TemplateOwner**
  - **TemplateCreator**
  - 目标集群的**ClusterAdmin**

警告

使用最少权限：在安装过程中，避免使用带有 RHV **SuperUser** 权限的管理员帐户。安装程序会将您提供的凭据保存到一个临时的 **ovirt-config.yaml** 文件中，这个凭证有被受到破坏的可能。

10.2.4. 验证 RHV 环境的要求

验证 RHV 环境是否满足安装和运行 OpenShift Container Platform 集群的要求。不满足这些要求会导致问题。



重要

这些要求基于安装程序用来创建 control plane 和计算机器的默认资源。这些资源包括 vCPU、内存和存储。如果更改这些资源或增加 OpenShift Container Platform 机器的数量，请相应调整这些要求。

流程

1. 检查 RHV 版本。
   a. 在 RHV 管理门户中，点右上角的 **?** 帮助图表，选 **About**。
   b. 在打开的窗口中，记录下 **RHV 软件版本**。

c. 确认 OpenShift Container Platform 版本 4.6 和您记录的 RHV 版本组合是被支持的。 RHV 上支持的 OpenShift Container Platform。

2. 检查数据中心、集群和存储。

a. 在 RHV 管理门户中，点 **Compute → Data Centers**。

b. 确认可以访问您要安装 OpenShift Container Platform 的数据中心。

c. 点击该数据中心的名称。

d. 在数据中心详情中，**存储** 标签中确认您要安装 OpenShift Container Platform 的存储域是 **Active**。

e. 记录下**域名**以供以后使用。

f. 确认 **Free Space** 至少为 230 GiB。

g. 确认存储域满足 etcd 后端性能要求，可以使用 fio 性能基准工具来评测。

h. 在数据中心详情中点击 **Clusters** 选项卡。

i. 找到您要安装 OpenShift Container Platform 的 RHV 集群。记录集群名称，以供稍后使用。

3. 检查 RHV 主机资源。

a. 在 RHV 管理门户中，点 **Compute > Clusters**。

b. 点击要安装 OpenShift Container Platform 的集群。

c. 在集群详情中点击 **Hosts** 标签页。

d. 检查主机，确认这些主机有至少 28 个 **逻辑 CPU 内核**，专门用于 OpenShift Container Platform 集群。

e. 记录**逻辑 CPU 内核**数以供稍后使用。

f. 请确认这些 CPU 内核被正确分配，在安装过程中创建的七台虚拟机中的每一台都可以有四个内核。

g. 确认主机总共有 112 GiB 的**Max free Memory for scheduling new virtual machines**，以满足以下每个 OpenShift Container Platform 机器的要求：

- bootstrap 机器需要 16 GiB
- 三个 control plane 机器每个机器都需要 16 GiB
- 三个计算机器每个机器都需要 16 GiB

h. 记录下 **Max free Memory for scheduling new virtual machine**的值以便稍后使用。

4. 验证安装 OpenShift Container Platform 的虚拟网络能否访问 RHV Manager 的 REST API。在这个网络的虚拟机上，使用 curl 来访问 RHV Manager 的 REST API:

```
$ curl -k -u <username>@<profile>:<password> \ 1
https://<engine-fqdn>/ovirt-engine/api 2
```

1. 对于 **<username>**，指定具有在 RHV 上创建和管理 OpenShift Container Platform 集群的 RHV 帐户的用户名。对于 **<profile>**，请指定登录配置集，您可以登陆到 RHV 管理门户查看 **Profile** 下拉列表。对于 **<password>**，指定该用户的密码。

2. 对于 **<engine-fqdn>**，请指定 RHV 环境的完全限定域名。

例如：

```
$ curl -k -u ocpadmin@internal:pw123 \
https://rhv-env.virtlab.example.com/ovirt-engine/api
```

10.2.5. 在 RHV 中准备网络环境

为 OpenShift Container Platform 集群配置两个静态 IP 地址，并使用这些地址创建 DNS 条目。

流程

1. 保留两个静态 IP 地址

   a. 在您要安装 OpenShift Container Platform 的网络上，标识 DHCP 租期池之外的两个静态 IP 地址。

   b. 连接到此网络中的主机，并确认每个 IP 地址都没有被使用。例如，使用地址解析协议（ARP）检查 IP 地址是否有条目：

   ```
   $ arp 10.35.1.19
   ```

   输出示例

   ```
   10.35.1.19 (10.35.1.19) -- no entry
   ```

   c. 为您的网络环境保留两个静态 IP 地址。

   d. 记录这些 IP 地址以备将来参考。

2. 为 OpenShift Container Platform REST API 创建 DNS 条目，并使用以下格式应用域名：

```
api.<cluster-name>.<base-domain>   <ip-address> 1
*.apps.<cluster-name>.<base-domain>   <ip-address> 2
```

1. 对于 **<cluster-name>**、**<base-domain>**和 **<ip-address>**，请指定 OpenShift Container Platform API 的集群名称、基域和静态 IP 地址。

2. 指定 Ingress 和负载均衡器的 OpenShift Container Platform 应用程序的集群名称、基域和静态 IP 地址。

例如：

```
api.my-cluster.virtlab.example.com 10.35.1.19
*.apps.my-cluster.virtlab.example.com 10.35.1.20
```

10.2.6. 为 RHV 设置 CA 证书

从 Red Hat Virtualization (RHV) Manager 下载 CA 证书，并在安装机器中进行设置。

您可以使用 RHV Manager 的网页或使用 **curl** 命令下载该证书。

之后，您向安装程序提供证书。

流程

1. 使用这两个方法之一下载 CA 证书：
   - 进入 Manager 的网页 **https://<engine-fqdn>/ovirt-engine/**。然后在 下载 中点击 CA 证书 链接。
   - 运行以下命令：

     ```
     $ curl -k 'https://<engine-fqdn>/ovirt-engine/services/pki-resource?resource=ca-certificate&format=X509-PEM-CA' -o /tmp/ca.pem
     ```
     (1)

     (1) 对于 **<engine-fqdn>**，请指定 RHV Manager 的全限定域名，如 **rhv-env.virtlab.example.com**。

2. 配置 CA 文件，为 Manager 授予无根用户访问权限。将 CA 文件权限设置为 **0644** （symbolic 值： **-rw-r—r--**）：

   ```
   $ sudo chmod 0644 /tmp/ca.pem
   ```

3. 对于 Linux，将 CA 证书复制到服务器证书目录中。使用 **-p** 保留权限：

   ```
   $ sudo cp -p /tmp/ca.pem /etc/pki/ca-trust/source/anchors/ca.pem
   ```

4. 将证书添加到您操作系统的证书管理器：
   - 对于 macOS，请双击这个证书文件，并使用 **Keychain Access** 程序将该文件添加到 **System** 密钥链中。
   - 对于 Linux，更新 CA 信任：

     ```
     $ sudo update-ca-trust
     ```

     注意

     如果使用您自己的证书认证机构，请确定系统信任它。

其他资源

- 如需了解更多相关信息，请参阅 RHV 文档中的 身份验证及安全性。

10.2.7. 生成 SSH 私钥并将其添加到代理中

如果要在集群上执行安装调试或灾难恢复，则必须为 **ssh-agent** 和安装程序提供 SSH 密钥。您可以使用此密钥访问公共集群中的 bootstrap 机器来排除安装问题。

注意

在生产环境中，您需要进行灾难恢复和调试。

您可以使用此密钥以 **core** 用户身份通过 SSH 连接到 master 节点。在部署集群时，此密钥会添加到 **core** 用户的 **~/.ssh/authorized_keys** 列表中。

流程

1. 如果还没有为计算机上免密码身份验证而配置的 SSH 密钥，请创建一个。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ ssh-keygen -t ed25519 -N '' \
  -f <path>/<file_name> 1
```

1 指定新 SSH 密钥的路径和文件名，如 **~/.ssh/id_rsa**。如果您已有密钥对，请确保您的公钥位于 **~/.ssh** 目录中。

运行此命令会在指定的位置生成不需要密码的 SSH 密钥。

注意

如果您计划在 **x86_64** 架构中安装使用 FIPS 验证的/Modules in Process 加密库的 OpenShift Container Platform 集群，不要创建使用 **ed25519** 算法的密钥。反之，创建一个使用 **rsa** 或 **ecdsa** 算法的密钥。

2. 作为后台任务启动 **ssh-agent** 进程：

```
$ eval "$(ssh-agent -s)"
```

输出示例

```
Agent pid 31874
```

注意

如果您的集群采用 FIPS 模式，则只使用 FIPS 兼容算法来生成 SSH 密钥。密钥必须是 RSA 或 ECDSA。

3. 将 SSH 私钥添加到 **ssh-agent**：

```
$ ssh-add <path>/<file_name> 1
```

输出示例

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

1 指定 SSH 私钥的路径和文件名，如 **~/.ssh/id_rsa**

后续步骤

- 在安装 OpenShift Container Platform 时，为安装程序提供 SSH 公钥。

10.2.8. 获取安装程序

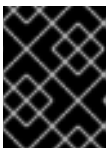
在安装 OpenShift Container Platform 之前，将安装文件下载到本地计算机上。

先决条件

- 运行 Linux 或 macOS 的计算机，本地磁盘空间为 500 MB

流程

1. 访问 OpenShift Cluster Manager 站点的 Infrastructure Provider 页面。如果您有红帽帐号，请使用自己的凭证登录。如果没有，请创建一个帐户。
2. 选择您的基础架构供应商。
3. 进入适用于您的安装类型的页面，下载您的操作系统的安装程序，并将文件放在要保存安装配置文件的目录中。。



重要

安装程序会在用来安装集群的计算机上创建若干文件。在完成集群安装后，您必须保留安装程序和安装程序所创建的文件。这两个文件都需要删除集群。

重要

删除安装程序创建的文件不会删除您的集群，即使集群在安装过程中失败也是如此。要删除集群，为特定云供应商完成 OpenShift Container Platform 卸载流程。

4. 提取安装程序。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ tar xvf openshift-install-linux.tar.gz
```

5. 从 Red Hat OpenShift Cluster Manager 下载安装 pull secret 。通过此 pull secret，您可以进行所含授权机构提供的服务的身份验证，这些服务包括为 OpenShift Container Platform 组件提供容器镜像的 Quay.io。

10.2.9. 创建安装配置文件

您可以自定义在 Red Hat Virtualization (RHV) 上安装的 OpenShift Container Platform 集群。

先决条件

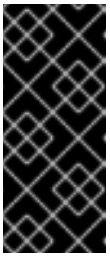
- 获取 OpenShift Container Platform 安装程序以及集群的 pull secret。

流程

1. 创建 **install-config.yaml** 文件。
   a. 更改到包含安装程序的目录，再运行以下命令：

```
$ ./openshift-install create install-config --dir <installation_directory> 1
```

1 对于 **<installation_directory>**，请指定用于保存安装程序所创建的文件的目录名称。



重要

指定一个空目录。一些安装信息，如 bootstrap X.509 证书，有较短的过期间隔，因此不要重复使用安装目录。如果要重复使用另一个集群安装中的个别文件，可以将其复制到您的目录中。但是，一些安装数据的文件名可能会在发行版本之间有所改变。从 OpenShift Container Platform 老版本中复制安装文件时要格外小心。

b. 根据安装程序提示输入相关的值。

i. 对于 **SSH 公钥**，请选择免密码公钥，如 **~/.ssh/id_rsa.pub**。这个密钥被用来验证与新的 OpenShift Container Platform 集群的连接。

注意

如果您要在生产环境中执行安装调试或灾难恢复，请指定 **ssh-agent** 进程需要使用的 SSH 密钥。

ii. 对于 **Platform**，选择 **ovirt**。

iii. 对于 **Enter oVirt's API endpoint URL**，使用以下格式输入 RHV API 的 URL：

```
https://<engine-fqdn>/ovirt-engine/api 1
```

1 对于 **<engine-fqdn>**，请指定 RHV 环境的完全限定域名。

例如：

```
$ curl -k -u ocpadmin@internal:pw123 \
https://rhv-env.virtlab.example.com/ovirt-engine/api
```

iv. 对于 **Is the oVirt CA local?**，输入 **Yes**，因为您已经设置了一个 CA 证书。否则，请输入 **No**。

v. 对于 **oVirt 的 CA bundle**，如果您为前一个问题输入的是 **Yes**，请从 **/etc/pki/ca-trust/source/anchors/ca.pem** 中复制证书内容并粘贴到这里。然后按两次 **Enter** 键。如果您就前一个问题输入 **No**，则不会出现这个问题。

vi. 对于 **oVirt engine username**，请使用以下格式输入 RHV 管理员的用户名和配置文件：

```
<username>@<profile> 1
```

1 对于 **<username>**，请指定 RHV 管理员的用户名。对于 **<profile>**，请指定登录配置集，您可以登陆到 RHV 管理门户查看 **Profile** 下拉列表。用户名和配置集应与以下示例相似：

```
ocpadmin@internal
```

vii. 对于 **oVirt engine password**，请输入 RHV admin 密码。

viii. 对于 **oVirt cluster**，请选择用于安装 OpenShift Container Platform 的集群。

ix. 对于 **oVirt storage domain**，请选择安装 OpenShift Container Platform 的存储域。

x. 对于 **oVirt network**，请选择可访问 Manager REST API 的虚拟网络。

xi. 对于 **Internal API Virtual IP**，请为集群的 REST API 输入您设置的静态 IP 地址。

xii. 对于 **Ingress virtual IP**，请为通配符应用程序域输入您保留的静态 IP 地址。

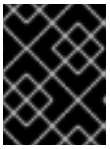
xiii. 对于 **Base Domain**，请输入 OpenShift Container Platform 集群的基域。如果这个群集暴露于外部世界，这必须是 DNS 基础结构可识别的有效域。例如： 输入 **virtlab.example.com**

xiv. 对于 **Cluster Name**，请输入集群名称。例如： **my-cluster**。使用您为 OpenShift Container Platform REST API 创建的外部注册/可解析 DNS 条目的集群名称，以及应用域名。安装程序也将此名称提供给 RHV 环境中的集群。

xv. 对于 **Pull Secret**，请从之前下载并粘贴的 **pull-secret.txt** 文件中复制 pull secret。您还可以从 Red Hat OpenShift Cluster Manager 获取同一 pull secret 的副本。

2. 修改 **install-config.yaml** 文件。您可以在**安装配置参数**部分中找到有关可用参数的更多信息。

3. 备份 **install-config.yaml** 文件，以便用于安装多个集群。



重要

install-config.yaml 文件会在安装过程中消耗掉。如果要重复使用此文件，必须现在备份。

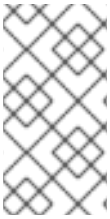
10.2.9.1. Red Hat Virtualization（RHV）的 install-config.yaml 文件示例

您可以通过更改 **install-config.yaml** 文件中的参数和参数值来自定义安装程序创建的 OpenShift Container Platform 集群。

以下示例专用于在 RHV 上安装 OpenShift Container Platform。

该文件位于您运行以下命令时指定的 **<installation_directory>** 中。

```
$ ./openshift-install create install-config --dir <installation_directory>
```



注意

- 这些示例文件仅供参考。您必须使用安装程序来获取 **install-config.yaml** 文件。
- 更改 **install-config.yaml** 文件可以增加集群所需的资源。验证您的 RHV 环境是否具有这些其他资源。否则，安装或集群将失败。

示例：这是默认的 install-config.yaml 文件

```
apiVersion: v1
baseDomain: example.com
compute:
- architecture: amd64
  hyperthreading: Enabled
  name: worker
  platform: {}
  replicas: 3
controlPlane:
  architecture: amd64
  hyperthreading: Enabled
  name: master
  platform: {}
  replicas: 3
metadata:
  creationTimestamp: null
  name: my-cluster
networking:
  clusterNetwork:
  - cidr: 10.128.0.0/14
    hostPrefix: 23
  machineNetwork:
  - cidr: 10.0.0.0/16
  networkType: OpenShiftSDN
  serviceNetwork:
  - 172.30.0.0/16
platform:
  ovirt:
    api_vip: 10.46.8.230
    ingress_vip: 192.168.1.5
    ovirt_cluster_id: 68833f9f-e89c-4891-b768-e2ba0815b76b
    ovirt_storage_domain_id: ed7b0f4e-0e96-492a-8fff-279213ee1468
    ovirt_network_name: ovirtmgmt
    vnicProfileID: 3fa86930-0be5-4052-b667-b79f0a729692
publish: External
pullSecret: '{"auths": ...}'
sshKey: ssh-ed12345 AAAA...
```

示例：最小 install-config.yaml 文件

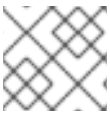
```
apiVersion: v1
baseDomain: example.com
metadata:
  name: test-cluster
platform:
  ovirt:
    api_vip: 10.46.8.230
    ingress_vip: 10.46.8.232
    ovirt_cluster_id: 68833f9f-e89c-4891-b768-e2ba0815b76b
    ovirt_storage_domain_id: ed7b0f4e-0e96-492a-8fff-279213ee1468
    ovirt_network_name: ovirtmgmt
    vnicProfileID: 3fa86930-0be5-4052-b667-b79f0a729692
pullSecret: '{"auths": ...}'
sshKey: ssh-ed12345 AAAA...
```

示例： install-config.yaml 文件中的自定义机器池

```
apiVersion: v1
baseDomain: example.com
controlPlane:
  name: master
  platform:
    ovirt:
      cpu:
        cores: 4
        sockets: 2
      memoryMB: 65536
      osDisk:
        sizeGB: 100
      vmType: server
  replicas: 3
compute:
- name: worker
  platform:
    ovirt:
      cpu:
        cores: 4
        sockets: 4
      memoryMB: 65536
      osDisk:
        sizeGB: 200
      vmType: server
  replicas: 5
metadata:
  name: test-cluster
platform:
  ovirt:
    api_vip: 10.46.8.230
    ingress_vip: 10.46.8.232
    ovirt_cluster_id: 68833f9f-e89c-4891-b768-e2ba0815b76b
    ovirt_storage_domain_id: ed7b0f4e-0e96-492a-8fff-279213ee1468
    ovirt_network_name: ovirtmgmt
    vnicProfileID: 3fa86930-0be5-4052-b667-b79f0a729692
pullSecret: '{"auths": ...}'
sshKey: ssh-ed25519 AAAA...
```

10.2.9.2. 安装配置参数

在部署 OpenShift Container Platform 集群前，您可以提供参数值，以描述托管集群的云平台的帐户并选择性地自定义集群平台。在创建 **install-config.yaml** 安装配置文件时，您可以通过命令行来提供所需的参数的值。如果要自定义集群，可以修改 **install-config.yaml** 文件来提供关于平台的更多信息。



注意

安装之后，您无法修改 **install-config.yaml** 文件中的这些参数。

重要

openshift-install 命令不验证参数的字段名称。如果指定了不正确的名称，则不会创建相关的文件或对象，且不会报告错误。确保所有指定的参数的字段名称都正确。

10.2.9.2.1. 所需的配置参数

下表描述了所需的安装配置参数：

表 10.1. 所需的参数

参数	描述	值
apiVersion	**install-config.yaml** 内容的 API 版本。当前版本是 **v1**。安装程序还可能支持旧的 API 版本。	字符串
baseDomain	云供应商的基域。此基础域用于创建到 OpenShift Container Platform 集群组件的路由。集群的完整 DNS 名称是 **baseDomain** 和 **metadata.name** 参数值的组合，其格式为 **<metadata.name>.<baseDomain>**。	完全限定域名或子域名，如 **example.com**。
metadata	Kubernetes 资源 **ObjectMeta**，其中只消耗 **name** 参数。	对象
metadata.name	集群的名称。集群的 DNS 记录是 **{{.metadata.name}}.{{.baseDomain}}** 的子域。	小写字母,连字符(-)和句点(.)的字符串，如 **dev**。
platform	执行安装的具体平台配置：**aws**、**baremetal**、**azure**、**openstack**、**ovirt**、**vsphere**。有关 **platform.<platform>** 参数的额外信息，请参考下表来了解您的具体平台。	对象

参数	描述	值

pullSecret	从 Red Hat OpenShift Cluster Manager 获取 pull secret，验证从 Quay.io 等服务中下载 OpenShift Container Platform 组件的容器镜像。	<pre>{
		  "auths":{
		  "cloud.openshift.com":{
		  "auth":"b3Blb=",
		  "email":"you@example.com"
		  },
		  "quay.io":{
		  "auth":"b3Blb=",
		  "email":"you@example.com"
		  }
		  }
		}</pre>

10.2.9.2.2. 网络配置参数

您可以根据现有网络基础架构的要求自定义安装配置。例如，您可以扩展集群网络的 IP 地址块，或者提供不同于默认值的不同 IP 地址块。

只支持 IPv4 地址。

表 10.2. 网络参数

参数	描述	值
networking	集群网络的配置。	对象 **注意** 您不能在安装后修改 **networking** 对象指定的参数。
networking.networkType	要安装的集群网络供应商 Container Network Interface（CNI）插件。	**OpenShiftSDN** 或 **OVNKubernetes**。默认值为 **OpenShiftSDN**。

networking.clusterNetwork	pod 的 IP 地址块。
	
	默认值为 **10.128.0.0/14**，主机前缀为 **/23**。
	
	如果您指定多个 IP 地址块，则块不得互相重叠。
		一个对象数组。例如：
		
		<pre>networking:
		  clusterNetwork:
		  - cidr: 10.128.0.0/14
		  hostPrefix: 23</pre>

networking.clusterNetwork.cidr	使用 **networking.clusterNetwork** 时需要此项。IP 地址块。 一个 IPv4 网络。	使用 CIDR 形式的 IP 地址块。IPv4 块的前缀长度介于 **0** 到 **32** 之间。

参数	描述	值
networking.clusterNetwork.hostPrefix	分配给每个单独节点的子网前缀长度。例如，如果 **hostPrefix** 设为 **23**，则每个节点从所给的 **cidr** 中分配一个 **/23** 子网。**hostPrefix** 值 **23** 提供 510（2^(32 - 23)- 2）个 pod IP 地址。	子网前缀。 默认值为 **23**。
networking.serviceNetwork	服务的 IP 地址块。默认值为 **172.30.0.0/16**。 OpenShift SDN 和 OVN-Kubernetes 网络供应商只支持服务网络的一个 IP 地址块。	CIDR 格式具有 IP 地址块的数组。例如： networking: serviceNetwork: - 172.30.0.0/16
networking.machineNetwork	机器的 IP 地址块。 如果您指定多个 IP 地址块，则块不得互相重叠。	一个对象数组。例如： networking: machineNetwork: - cidr: 10.0.0.0/16
networking.machineNetwork.cidr	使用 **networking.machineNetwork** 时需要。IP 地址块。libvirt 以外的所有平台的默认值为 **10.0.0.0/16**。对于 libvirt，默认值为 **192.168.126.0/24**。	CIDR 表示法中的 IP 网络块。 例如： **10.0.0.0/16**。 **注意** 将 **networking.machineNetwork** 设置为与首选 NIC 所在的 CIDR 匹配。

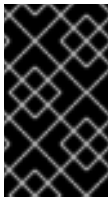
10.2.9.2.3. 可选配置参数

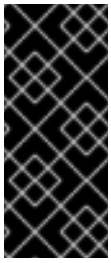
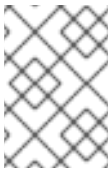
下表描述了可选安装配置参数：

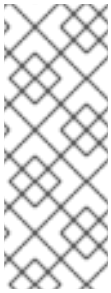
表 10.3. 可选参数

参数	描述	值
additionalTrustBundle	添加到节点可信证书存储中的 PEM 编码 X.509 证书捆绑包。配置了代理时，也可以使用这个信任捆绑包。	字符串
compute	组成计算节点的机器的配置。	machine-pool 对象的数组。详情请查看以下"Machine-pool"表。

参数	描述	值
compute.architecture	决定池中机器的指令集合架构。目前不支持异构集群，因此所有池都必须指定相同的架构。有效值为 **amd64** （默认值）。	字符串
compute.hyperthreading	是否在计算机器上启用或禁用并发多线程或**超线程**。默认情况下，启用并发多线程以提高机器内核的性能。  **重要** 如果禁用并发多线程，请确保在容量规划时考虑到机器性能可能会显著降低的问题。	**Enabled** 或 **Disabled**
compute.name	使用 **compute** 时需要此值。机器池的名称。	**worker**
compute.platform	使用 **compute** 时需要此值。使用此参数指定托管 worker 机器的云供应商。此参数值必须与 **controlPlane.platform** 参数值匹配。	**aws**、**azure**、**gcp**、**openstack**、**ovirt**、**vsphere** 或 **{}**
compute.replicas	要置备的计算机器数量，也称为 worker 机器。	大于或等于 **2** 的正整数。默认值为 **3**。
controlPlane	组成 control plane 的机器的配置。	**MachinePool** 对象的数组。详情请查看以下"Machine-pool"表。
controlPlane.architecture	决定池中机器的指令集合架构。目前不支持异构集群，因此所有池都必须指定相同的架构。有效值为 **amd64** （默认值）。	字符串

参数	描述	值
controlPlane.hyperthreading	是否在 control plane 机器上启用或禁用并发多线程或**超线程**。默认情况下，启用并发多线程以提高机器内核的性能。  **重要** 如果禁用并发多线程，请确保在容量规划时考虑到机器性能可能会显著降低的问题。	**Enabled** 或 **Disabled**
controlPlane.name	使用 **controlPlane** 时需要。机器池的名称。	**master**
controlPlane.platform	使用 **controlPlane** 时需要。使用此参数指定托管 control plane 机器的云供应商。此参数值必须与 **compute.platform** 参数值匹配。	**aws**、**azure**、**gcp**、**openstack**、**ovirt**、**vsphere** 或 **{}**
controlPlane.replicas	要置备的 control plane 机器数量。	唯一支持的值是 **3**，它是默认值。
credentialsMode	Cloud Credential Operator（CCO）模式。如果没有指定任何模式，CCO 会动态地尝试决定提供的凭证的功能，在支持多个模式的平台上使用 mint 模式。  **注意** 不是所有 CCO 模式都支持所有云供应商。如需有关 CCO 模式的更多信息，请参阅*Red Hat Operator 参考指南*内容中的 *Cloud Credential Operator* 条目。	**Mint**、**Passthrough**、**Manual** 或空字符串(**""**)。

参数	描述	值
fips	启用或禁用 FIPS 模式。默认为 **false**（禁用）。如果启用了 FIPS 模式，运行 OpenShift Container Platform 的 Red Hat Enterprise Linux CoreOS (RHCOS) 机器会绕过默认的 Kubernetes 加密套件，并使用由 RHCOS 提供的加密模块。  **重要** 只有在 **x86_64** 架构中的 OpenShift Container Platform 部署支持 FIPS 验证的/Modules in Process 加密库。  **注意** 如果使用 Azure File 存储，则无法启用 FIPS 模式。	**false** 或 **true**
imageContentSources	release-image 内容的源和仓库。	对象数组。包括一个 **source** 以及可选的 **mirrors**，如下表所示。
imageContentSources.source	使用 **imageContentSources** 时需要。指定用户在镜像拉取规格中引用的仓库。	字符串
imageContentSources.mirrors	指定可能还包含同一镜像的一个或多个仓库。	字符串数组
publish	如何发布或公开集群的面向用户的端点，如 Kubernetes API、OpenShift 路由。	**Internal** 或 **External**。默认值为 **External**。 在非云平台上不支持将此字段设置为 **Internal**。  **重要** 如果将字段的值设为 **Internal**，集群将无法运行。如需更多信息，请参阅 BZ#1953035。

参数	描述	值
sshKey	用于验证集群机器访问的 SSH 密钥或密钥。  **注意** 对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 **ssh-agent** 进程使用的 SSH 密钥。	一个或多个密钥。例如： sshKey: <key1> <key2> <key3>

10.2.9.2.4. 其他 Red Hat Virtualization（RHV）配置参数

下表描述了额外的 RHV 配置参数：

表 10.4. 集群的其他 RHV 参数

参数	描述	值
platform.ovirt.ovirt_cluster_id	必需。创建虚拟机的集群。	字符串。例如：68833 **f9f-e89c-4891-b768-e2ba0815b76b**
platform.ovirt.ovirt_storage_domain_id	必需。创建虚拟机磁盘的存储域 ID。	字符串。例如：**ed7b0f4e-0e96-492a-8fff-279213ee1468**
platform.ovirt.ovirt_network_name	必需。创建 VM nics 的网络名称。	字符串。例如：**ocpcluster**
platform.ovirt.vnicProfileID	必需。VM 网络接口的 vNIC 配置集 ID。如果集群网络只有一个配置集，则可以推断出这个值。	字符串。例如：**3fa86930-0be5-4052-b667-b79f0a729692**
platform.ovirt.api_vip	必需。分配给 API 虚拟 IP（VIP）的机器网络上的 IP 地址。您可以在此端点访问 OpenShift API。	字符串。示例：**10.46.8.230**
platform.ovirt.ingress_vip	必需。分配给 Ingress 虚拟 IP（VIP）的机器网络上的 IP 地址。	字符串。示例：**10.46.8.232**

10.2.9.2.5. 机器池的其他 RHV 参数

下表描述了机器池的其他 RHV 配置参数：

表 10.5. 机器池的其他 RHV 参数

参数	描述	值
<machine-pool>.platform.ovirt.cpu	可选。定义虚拟机的 CPU。	对象
<machine-pool>.platform.ovirt.cpu.cores	使用 **<machine-pool>.platform.ovirt.cpu** 时需要此项。内核数。虚拟 CPU 总数（vCPU）是内核 * 插槽。	整数
<machine-pool>.platform.ovirt.cpu.sockets	使用 **<machine-pool>.platform.ovirt.cpu** 时需要此项。每个内核的插槽数。虚拟 CPU 总数（vCPU）是内核 * 插槽。	整数
<machine-pool>.platform.ovirt.memoryMB	可选。MiB 中虚拟机的内存。	整数
<machine-pool>.platform.ovirt.instanceTypeID	可选。一个实例类型 UUID，如 **00000009-0009-0009-0009-0000000000f1**，它可以从 **https://<engine-fqdn>/ovirt-engine/api/instancetypes** 端点获得。	UUID 字符串
<machine-pool>.platform.ovirt.osDisk	可选。定义虚拟机的第一个和可引导磁盘。	字符串
<machine-pool>.platform.ovirt.osDisk.sizeGB	如果您使用 **<machine-pool>.platform.ovirt.osDisk**，则需要此操作。GiB 中磁盘的大小。	数字
<machine-pool>.platform.ovirt.vmType	可选。VM 工作负载类型，如 **high-performance**、**server** 或 **desktop**。	字符串

注意

您可以将 **<machine-pool>** 替换为 **controlPlane** 或 **compute**。

10.2.10. 部署集群

您可以在兼容云平台中安装 OpenShift Container Platform。

重要

安装程序的 **create cluster** 命令只能在初始安装过程中运行一次。

先决条件

- 从运行安装程序的机器中打开到 Manager 的 **ovirt-imageio** 端口。默认情况下，端口为 **54322**。
- 获取 OpenShift Container Platform 安装程序以及集群的 pull secret。

流程

1. 更改为包含安装程序的目录并初始化集群部署：

```
$ ./openshift-install create cluster --dir <installation_directory> \ 1
    --log-level=info 2
```

1 对于 **<installation_directory>**，请指定自定义 **./install-config.yaml** 文件的位置。

2 要查看不同的安装详情，请指定 **warn**、**debug** 或 **error**，而不要指定 **info**。

注意

如果您在主机上配置的云供应商帐户没有足够的权限来部署集群，安装过程将会停止，并且显示缺少的权限。

集群部署完成后，终端会显示访问集群的信息，包括指向其 Web 控制台的链接和 **kubeadmin** 用户的凭证。

输出示例

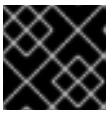
```
...
INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run 'export
KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'
INFO Access the OpenShift web-console here: https://console-openshift-
console.apps.mycluster.example.com
INFO Login to the console with user: "kubeadmin", and password: "4vYBz-Ee6gm-ymBZj-
Wt5AL"
INFO Time elapsed: 36m22s
```

注意

当安装成功时，集群访问和凭证信息还会输出到 **<installation_directory>/.openshift_install.log**。

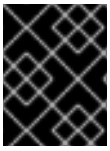
重要

- 安装程序生成的 Ignition 配置文件包含在 24 小时后过期的证书，然后在过期时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外情况是，您需要手动批准待处理的 **node-bootstrapper** 证书签名请求（CSR）来恢复 kubelet 证书。如需更多信息，请参阅*从过期的 control plane 证书中恢复*的文档。
- 建议您在生成 12 小时后使用 Ignition 配置文件，因为集群安装后 24 小时证书从 16 小时轮转至 22 小时。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中运行证书更新时避免安装失败。



重要

您不得删除安装程序或安装程序所创建的文件。需要这两者才能删除集群。



重要

您已完成了安装集群所需的步骤。余下的步骤演示了如何验证集群并对安装进行故障排除。

10.2.11. 通过下载二进制文件安装 OpenShift CLI

您需要安装 CLI（**oc**） 来使用命令行界面与 OpenShift Container Platform 进行交互。您可在 Linux 、Windows 或 macOS 上安装 **oc**。

重要

如果安装了旧版本的 **oc**，则无法使用 OpenShift Container Platform 4.6 中的所有命令。下载并安装新版本的 **oc**。

10.2.11.1. 在 Linux 上安装 OpenShift CLI

您可以按照以下流程在 Linux 上安装 OpenShift CLI（**oc**）二进制文件。

流程

1. 进入到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 在 **Version** 下拉菜单中选择相应的版本。
3. 单击 **OpenShift v4.6 Linux 客户端** 条目旁边的 **Download Now**，再保存文件。
4. 解包存档：

```
$ tar xvzf <file>
```

5. 把 **oc** 二进制代码放到 **PATH** 中的目录中。
   执行以下命令可以查看当前的 **PATH** 设置：

```
$ echo $PATH
```

安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
$ oc <command>
```

10.2.11.2. 在 Windows 上安装 OpenShift CLI

您可以按照以下流程在 Windows 上安装 OpenShift CLI（**oc**）二进制代码。

流程

1. 进入到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 在 **Version** 下拉菜单中选择相应的版本。

3. 单击 **OpenShift v4.6 Windows 客户端**条目旁边的 **Download Now**，再保存文件。
4. 使用 ZIP 程序解压存档。
5. 把 **oc** 二进制代码放到 **PATH** 中的目录中。
   要查看您的 **PATH**，请打开命令提示窗口并执行以下命令：

```
C:\> path
```

安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
C:\> oc <command>
```

10.2.11.3. 在 macOS 上安装 OpenShift CLI

您可以按照以下流程在 macOS 上安装 OpenShift CLI（**oc**）二进制代码。

流程

1. 进入到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 在 **Version** 下拉菜单中选择相应的版本。
3. 单击 **OpenShift v4.6 MacOSX 客户端**条目旁边的 **Download Now**，再保存文件。
4. 解包和解压存档。
5. 将 **oc** 二进制文件移到 PATH 的目录中。
   要查看您的 **PATH**，打开一个终端窗口并执行以下命令：

```
$ echo $PATH
```

安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
$ oc <command>
```

10.2.12. 使用 CLI 登录到集群

您可以通过导出集群 **kubeconfig** 文件，以默认系统用户身份登录集群。**kubeconfig** 文件包含关于集群的信息，供 CLI 用于将客户端连接到正确集群和 API 服务器。该文件特只适用于一个特定的集群，在 OpenShift Container Platform 安装过程中创建。

先决条件

- 已部署了 OpenShift Container Platform 集群。
- 已安装 **oc** CLI。

流程

1. 导出 **kubeadmin** 凭证：

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

2. 使用导出的配置，验证能否成功运行 **oc** 命令：

```
$ oc whoami
```

输出示例

```
system:admin
```

如需更多信息，请参阅 OpenShift CLI 入门。

10.2.13. 验证集群状态

您可以在安装过程中或安装后验证 OpenShift Container Platform 集群的状态：

流程

1. 在集群环境中，导出管理员的 kubeconfig 文件：

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

kubeconfig 文件包含关于集群的信息，供 CLI 用于将客户端连接到正确集群和 API 服务器。

2. 查看部署后创建的 control plane 和计算机器：

```
$ oc get nodes
```

3. 查看集群的版本：

```
$ oc get clusterversion
```

4. 查看 Operator 的状态：

```
$ oc get clusteroperator
```

5. 查看集群中的所有正在运行的 pod:

```
$ oc get pods -A
```

故障排除

如果安装失败，安装程序会超时并显示出错信息。如需了解更多相关信息，请参阅故障排除安装问题。

10.2.14. 访问 RHV 上的 OpenShift Container Platform Web 控制台。

OpenShift Container Platform 集群初始化后，您可以登录到 OpenShift Container Platform Web 控制台。

流程

1. 可选：在 Red Hat Virtualization (RHV) 管理门户中，打开 **Compute → Cluster**。
2. 验证安装程序是否创建了虚拟机。
3. 返回到安装程序正在运行的命令行。当安装程序完成后，它会显示登录到 OpenShift Container Platform Web 控制台的用户名和临时密码。
4. 在浏览器中，打开 OpenShift Container Platform web 控制台的 URL。URL 使用以下格式：

```
console-openshift-console.apps.<clustername>.<basedomain>
```
(1)

(1) 对于 **<clustername>.<baseDomain>**，请指定集群名称和基域。

例如：

```
console-openshift-console.apps.my-cluster.virtlab.example.com
```

10.2.15. OpenShift Container Platform 的 Telemetry 访问

在 OpenShift Container Platform 4.6 中，默认运行的 Telemetry 服务提供有关集群健康状况和成功更新的指标，需要访问互联网。如果您的集群连接到互联网，Telemetry 会自动运行，而且集群会注册到 OpenShift Cluster Manager。

确认 OpenShift Cluster Manager 清单正确后，可以由 Telemetry 自动维护，也可以使用 OpenShift Cluster Manager 手动维护，使用订阅监控来跟踪帐户或多集群级别的 OpenShift Container Platform 订阅。

其他资源

- 有关 Telemetry 服务的更多信息，请参阅关于远程健康监控。

10.2.16. 在 Red Hat Virtualization (RHV) 上安装时的常见问题

以下是您可能会遇到的一些常见问题，以及推荐的原因和解决方案。

10.2.16.1. CPU 负载增加和节点进入非就绪状态

- **症状**: CPU 负载显著增加，节点开始处于 **Not Ready** 状态。
- **原因**: 存储域延迟可能太大，特别是针对 control plane 节点（也称为 master 节点）。
- **解决方案**:
  通过重启 kubelet 服务使节点再次就绪：

```
$ systemctl restart kubelet
```

  检查 OpenShift Container Platform 指标服务，该服务可自动收集并报告一些重要数据，如 etcd 磁盘同步持续时间。如果集群是可操作的，使用这个数据来帮助确定这个问题是否是因为存储延迟或吞吐量造成的。如果是这样，请考虑使用一个较低延迟和更高吞吐量的存储资源。

  要获得原始指标，请以 kubeadmin 或具有 cluster-admin 特权的用户身份输入以下命令：

```
$ oc get --insecure-skip-tls-verify --server=https://localhost:<port> --raw=/metrics
```

如需了解更多相关信息，请参阅 使用 OpenShift 4.x 调试应用程序端点 。

10.2.16.2. 连接到 OpenShift Container Platform 集群 API 存在问题

- **症状**: 安装程序完成，但无法使用 OpenShift Container Platform 集群 API。在 bootstrap 过程完成后，bootstrap 虚拟机仍处于在线状态。当您输入以下命令时，回复会超时。

```
$ oc login -u kubeadmin -p *** <apiurl>
```

- **原因**: 安装程序没有删除 bootstrap VM，因此没有释放集群的 API IP 地址。
- **解决方法**：使用 **wait-for** 子命令，在 bootstrap 过程完成后获得通知：

```
$ ./openshift-install wait-for bootstrap-complete
```

当 bootstrap 过程完成后，删除 bootstrap 虚拟机：

```
$ ./openshift-install destroy bootstrap
```

10.2.17. 安装后的任务

在 OpenShift Container Platform 集群初始化后，您可以执行以下任务。

- 可选： 在部署后，使用 OpenShift Container Platform 中的 Machine Config Operator (MCO) 添加或替换 SSH 密钥。
- 可选：删除 **kubeadmin** 用户。使用身份验证提供程序创建具有 cluster-admin 权限的用户。

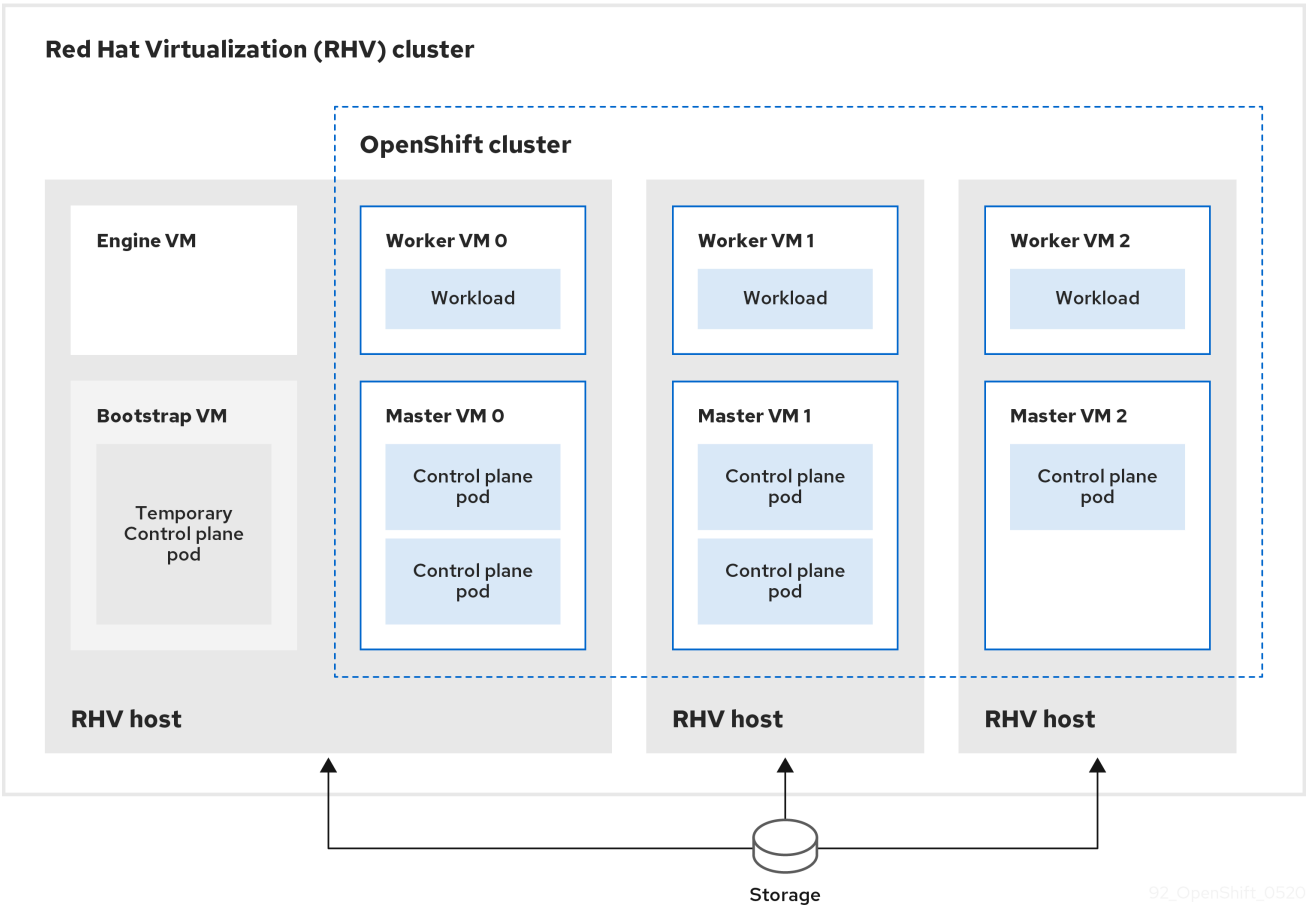
10.2.18. 后续步骤

- 自定义集群。
- 如果需要，您可以选择不使用远程健康报告。

10.3. 使用用户置备的基础架构在 RHV 上安装集群

在 OpenShift Container Platform 版本 4.6 中，您可以在 Red Hat Virtualization（RHV）和其他您提供的基础架构上安装自定义的 OpenShift Container Platform 集群。OpenShift Container Platform 文档使用*用户置备的基础架构*来引用此基础架构类型。

下图显示了在 RHV 集群上运行的潜在 OpenShift Container Platform 集群示例。



RHV 主机运行包含 control plane 和计算 pod 的虚拟机。其中一个主机还运行管理器虚拟机，以及包含临时 control plane pod 的 bootstrap 虚拟机。]

10.3.1. 先决条件

在 RHV 环境上安装 OpenShift Container Platform 集群需要满足以下条件。

- 在 Red Hat Virtualization(RHV)上的 OpenShift Container Platform Support Matrix 中支持的版本组合。
- 熟悉 OpenShift Container Platform 安装和更新流程。

10.3.2. OpenShift Container Platform 的互联网访问

在 OpenShift Container Platform 4.6 中，您需要访问互联网来安装集群。

您必须具有以下互联网访问权限：

- 访问 OpenShift Cluster Manager 以下载安装程序并执行订阅管理。如果集群可以访问互联网，并且没有禁用 Telemetry，该服务会自动授权您的集群。
- 访问 Quay.io，以获取安装集群所需的软件包。
- 获取执行集群更新所需的软件包。

重要

如果您的集群无法直接访问互联网，则可以在置备的某些类基础架构上执行受限网络安装。在此过程中，您要下载所需的内容，并使用它在镜像 registry（mirror registry） 中填充安装集群并生成安装程序所需的软件包。对于某些安装类型，集群要安装到的环境不需要访问互联网。在更新集群之前，要更新 registry 镜像系统中的内容。

10.3.3. RHV 环境的要求

要安装并运行 OpenShift Container Platform 集群，RHV 环境必须满足以下要求。

不满足这些要求会导致安装或进程失败。另外，无法满足这些要求可能会导致 OpenShift Container Platform 集群在安装后几天或几星期后失败。

对 CPU、内存和存储资源的以下要求是基于**默认值**乘以安装程序创建的默认虚拟机数。除了 RHV 环境用于非 OpenShift Container Platform 操作的资源外，**这些资源还必须可用**。

默认情况下，安装程序会在安装过程中创建七台虚拟机。首先，它会创建一个 bootstrap 虚拟机来提供临时服务和 control plane，同时创建 OpenShift Container Platform 集群的其余部分。当安装程序完成集群创建时，删除 bootstrap 机器可释放其资源。

如果在 RHV 环境中增加虚拟机数量，则需要相应地增加资源。

要求

- RHV 环境有一个数据中心，其状态是 **Up**。
- RHV 数据中心包含一个 RHV 集群。
- RHV 集群具有专门用于 OpenShift Container Platform 集群的以下资源：
  - 最小 28 个 vCPU：在安装过程中创建的七个虚拟机，每个都需要 4 个。
  - 112 GiB RAM 或更多，包括：
    - 16 GiB 或更多用于提供临时 control plane 功能的 bootstrap 机器。
    - 每个提供 control plane 功能的三台 control plane 机器都需要16 GiB 或更多。
    - 每个用来运行应用程序负载的 compute 机器都需要 16 GiB 或更多。
- RHV 存储域必须满足 etcd 后端性能要求。
- 在生产环境中，每个虚拟机必须具有 120 GiB 或更多存储。因此，存储域必须为默认的 OpenShift Container Platform 集群提供 840 GiB 或更多存储。在资源有限制或非生产环境中，每个虚拟机必须具有 32 GiB 或更多存储，因此对于默认的 OpenShift Container Platform 集群，存储域必须具有 230 GiB 或更多存储 。
- 要在安装和更新过程中从红帽生态系统目录下载镜像，RHV 集群必须可以访问互联网。Telemetry 服务还需要互联网连接来简化订阅和权利的过程。
- RHV 集群需要有一个虚拟网络，可访问 RHV Manager 上的 REST API。确保在这个网络中启用了 DHCP，因为安装程序创建的虚拟机会使用 DHCP 来获取他们的 IP 地址。
- 具有以下最少权限的用户帐户和组，用于在目标 RHV 集群上安装和管理 OpenShift Container Platform 集群：

- **DiskOperator**
- **DiskCreator**
- **UserTemplateBasedVm**
- **TemplateOwner**
- **TemplateCreator**
- 目标集群的**ClusterAdmin**

警告

使用最少权限：在安装过程中，避免使用带有 RHV **SuperUser** 权限的管理员帐户。安装程序会将您提供的凭据保存到一个临时的 **ovirt-config.yaml** 文件中，这个凭证有被受到破坏的可能。

10.3.4. 验证 RHV 环境的要求

验证 RHV 环境是否满足安装和运行 OpenShift Container Platform 集群的要求。不满足这些要求会导致问题。

重要

这些要求基于安装程序用来创建 control plane 和计算机器的默认资源。这些资源包括 vCPU、内存和存储。如果更改这些资源或增加 OpenShift Container Platform 机器的数量，请相应调整这些要求。

流程

1. 检查 RHV 版本。
   a. 在 RHV 管理门户中，点右上角的？帮助图表，选 **About**。
   b. 在打开的窗口中，记录下 **RHV 软件版本**。
   c. 确认 OpenShift Container Platform 版本 4.6 和您记录的 RHV 版本组合是被支持的。 RHV 上支持的 OpenShift Container Platform。
2. 检查数据中心、集群和存储。
   a. 在 RHV 管理门户中，点 **Compute → Data Centers**。
   b. 确认可以访问您要安装 OpenShift Container Platform 的数据中心。
   c. 点击该数据中心的名称。
   d. 在数据中心详情中，**存储** 标签中确认您要安装 OpenShift Container Platform 的存储域是 **Active**。
   e. 记录下**域名**以供以后使用。

f. 确认 **Free Space** 至少为 230 GiB。

g. 确认存储域满足 etcd 后端性能要求，可以使用 fio 性能基准工具来评测。

h. 在数据中心详情中点击 **Clusters** 选项卡。

i. 找到您要安装 OpenShift Container Platform 的 RHV 集群。记录集群名称，以供稍后使用。

3. 检查 RHV 主机资源。

a. 在 RHV 管理门户中，点 **Compute** > **Clusters**。

b. 点击要安装 OpenShift Container Platform 的集群。

c. 在集群详情中点击 **Hosts** 标签页。

d. 检查主机，确认这些主机有至少 28 个 **逻辑 CPU 内核**，专门用于 OpenShift Container Platform 集群。

e. 记录**逻辑 CPU 内核**数以供稍后使用。

f. 请确认这些 CPU 内核被正确分配，在安装过程中创建的七台虚拟机中的每一台都可以有四个内核。

g. 确认主机总共有 112 GiB 的**Max free Memory for scheduling new virtual machines**，以满足以下每个 OpenShift Container Platform 机器的要求：

- bootstrap 机器需要 16 GiB
- 三个 control plane 机器每个机器都需要 16 GiB
- 三个计算机器每个机器都需要 16 GiB

h. 记录下 **Max free Memory for scheduling new virtual machine**的值以便稍后使用。

4. 验证安装 OpenShift Container Platform 的虚拟网络能否访问 RHV Manager 的 REST API。在这个网络的虚拟机上，使用 curl 来访问 RHV Manager 的 REST API:

```
$ curl -k -u <username>@<profile>:<password> \ 1
https://<engine-fqdn>/ovirt-engine/api 2
```

1 对于 **<username>**，指定具有在 RHV 上创建和管理 OpenShift Container Platform 集群的 RHV 帐户的用户名。对于 **<profile>**，请指定登录配置集，您可以登陆到 RHV 管理门户查看 **Profile** 下拉列表。对于 **<password>**，指定该用户的密码。

2 对于 **<engine-fqdn>**，请指定 RHV 环境的完全限定域名。

例如：

```
$ curl -k -u ocpadmin@internal:pw123 \
https://rhv-env.virtlab.example.com/ovirt-engine/api
```

10.3.5. 用户置备的基础架构对网络的要求

所有 Red Hat Enterprise Linux CoreOS（RHCOS）机器在启动过程中需要 **initramfs** 中的网络从机器配置服务器获取 Ignition 配置。

在初次启动过程中，需要一个 DHCP 服务器或集群中的每个机器都设置了静态 IP 地址来建立网络连接，以下载它们的 Ignition 配置文件。

建议您使用 DHCP 服务器为集群进行长期机器管理。确保 DHCP 服务器已配置为向集群机器提供持久 IP 地址和主机名。

Kubernetes API 服务器必须能够解析集群机器的节点名称。如果 API 服务器和 worker 节点位于不同的区域中，您可以配置默认 DNS 搜索区域，以便 API 服务器能够解析节点名称。另一种支持的方法是始终在节点对象和所有 DNS 请求中使用完全限定域名来指代主机。

您必须配置机器间的网络连接，以便集群组件进行通信。每台机器都必须能够解析集群中所有其他机器的主机名。

防火墙

配置防火墙以便集群能够访问所需的站点。

另请参阅：

- Red Hat Virtualization Manager 防火墙要求
- 主机防火墙要求

负载均衡器

配置一个（最好为两个）L4 负载均衡器：

- 为 control-plane 和 bootstrap 机器上的端口 **6443** 和 **22623** 提供负载均衡。端口 **6443** 提供对 Kubernetes API 服务器的访问，且必须在内部和外部访问。集群内的节点必须能够访问端口 **22623**。
- 为运行入口路由器（通常是默认配置中的计算节点）的机器提供端口 **443** 和 **80** 的负载均衡。这两个端口都必须从集群内部和外部访问。

DNS

配置基础架构提供的 DNS 以便正确解析主要组件和服务。如果您只使用一个负载均衡器，这些 DNS 记录可以指向相同的 IP 地址。

- 为 **api.<cluster_name>.<base_domain>**（内部和外部解析）和 **api-int.<cluster_name>.<base_domain>**（内部解析）创建 DNS 记录，指向 control plane 机器的负载均衡器。
- 为 ***.apps.<cluster_name>.<base_domain>** 创建一个 DNS 记录，指向入口路由器的负载均衡器。例如，计算机器的端口 **443** 和 **80**。

表 10.6. 所有机器到所有机器

协议	端口	描述
ICMP	N/A	网络可访问性测试
TCP	**1936**	指标

协议	端口	描述
	9000-9999	主机级别的服务，包括端口 **9100-9101** 上的节点导出器和端口 **9099** 上的 Cluster Version Operator。
	10250-10259	Kubernetes 保留的默认端口
	10256	openshift-sdn
UDP	**4789**	VXLAN 和 Geneve
	6081	VXLAN 和 Geneve
	9000-9999	主机级别的服务，包括端口 **9100-9101** 上的节点导出器。
TCP/UDP	**30000-32767**	Kubernetes 节点端口

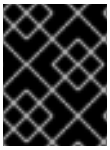
表 10.7. 要通过控制平面的所有机器

协议	端口	描述
TCP	**6443**	Kubernetes API

表 10.8. control plane 机器到 control plane 机器

协议	端口	描述
TCP	**2379-2380**	etcd 服务器和对等端口

网络拓扑要求
您为集群置备的基础架构必须满足下列网络拓扑要求。



重要

OpenShift Container Platform 要求所有节点都能访问互联网，以便为平台容器提取镜像并向红帽提供遥测数据。

负载均衡器
在安装 OpenShift Container Platform 前，您必须置备两个满足以下要求的负载均衡器：

1. **API 负载均衡器**：提供一个通用端点，供用户（包括人和机器）与平台交互和配置。配置以下条件：
   - 只适用于第 4 层负载均衡。这可被称为 Raw TCP、SSL Passthrough 或者 SSL 桥接模式。如果使用 SSL Bridge 模式，必须为 API 路由启用 Server Name Indication（SNI）。
   - 无状态负载平衡算法。这些选项根据负载均衡器的实现而有所不同。

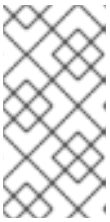
重要

不要为 API 负载均衡器配置会话持久性。

在负载均衡器的前端和后台配置以下端口：

表 10.9. API 负载均衡器

端口	后端机器（池成员）	内部	外部	描述
6443	Bootstrap 和 control plane.bootstrap 机器初始化集群 control plane 后，您要从负载均衡器中删除 bootstrap 机器。您必须为 API 服务器健康检查探测配置 **/readyz** 端点。	X	X	Kubernetes API 服务器
22623	Bootstrap 和 control plane.bootstrap 机器初始化集群 control plane 后，您要从负载均衡器中删除 bootstrap 机器。	X		机器配置服务器



注意

负载均衡器必须配置为，从 API 服务器关闭 **/readyz** 端点到从池中删除 API 服务器实例时最多需要 30 秒。在 **/readyz** 返回错误或处于健康状态后的时间范围内，端点必须被删除或添加。每 5 秒或 10 秒探测一次，有两个成功请求处于健康状态，三个成为不健康的请求经过测试。

2. **应用程序入口负载均衡器**:提供来自集群外部的应用程序流量流量的 Ingress 点。配置以下条件：

   - 只适用于第 4 层负载均衡。这可被称为 Raw TCP、SSL Passthrough 或者 SSL 桥接模式。如果使用 SSL Bridge 模式，您必须为 Ingress 路由启用Server Name Indication（SNI）。
   - 建议根据可用选项以及平台上托管的应用程序类型，使用基于连接的或者基于会话的持久性。

   在负载均衡器的前端和后台配置以下端口：

   表 10.10. 应用程序入口负载均衡器

端口	后端机器（池成员）	内部	外部	描述
443	默认运行入口路由器 Pod、计算或 worker 的机器。	X	X	HTTPS 流量
80	默认运行入口路由器 Pod、计算或 worker 的机器。	X	X	HTTP 流量

提示

如果负载均衡器可以看到客户端的真实 IP 地址，启用基于 IP 的会话持久性可提高使用端到端 TLS 加密的应用程序的性能。

注意

OpenShift Container Platform 集群需要正确配置入口路由器。control plane 初始化后，您必须配置入口路由器。

NTP 配置

OpenShift Container Platform 集群默认配置为使用公共网络时间协议（NTP）服务器。如果要使用本地企业 NTP 服务器，或者集群部署在断开连接的网络中，您可以将集群配置为使用特定的时间服务器。如需更多信息，请参阅*配置 chrony 时间服务*的文档。

如果 DHCP 服务器提供 NTP 服务器信息，Red Hat Enterprise Linux CoreOS（RHCOS）机器上的 chrony 时间服务会读取信息，并可与 NTP 服务器同步时钟。

10.3.6. 设置安装机器

要运行二进制 **openshift-install** 安装程序和 Ansible 脚本，请设置 RHV Manager 或具有网络访问 RHV 环境的 Red Hat Enterprise Linux (RHEL) 计算机以及 Manager 上的 REST API。

流程

1. 更新或安装 Python3 和 Ansible。例如：

```
# dnf update python3 ansible
```

2. 安装 **python3-ovirt-engine-sdk4** 软件包 来获取 Python 软件开发组件。

3. 安装 **ovirt.image-template** Ansible 角色。在 RHV Manager 和其他 Red Hat Enterprise Linux (RHEL) 机器上，这个角色作为 **ovirt-ansible-image-template** 软件包发布。例如，输入：

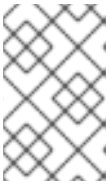
```
# dnf install ovirt-ansible-image-template
```

4. 安装 **ovirt.vm-infra** Ansible 角色。在 RHV Manager 和其他 RHEL 机器上，此角色作为 **ovirt-ansible-vm-infra** 软件包发布。

```
# dnf install ovirt-ansible-vm-infra
```

5. 创建环境变量并为其分配绝对或相对路径。例如，输入：

```
$ export ASSETS_DIR=./wrk
```



注意

安装程序使用这个变量创建保存重要安装相关文件的目录。之后，安装过程会重复使用此变量来定位这些资产文件。避免删除这个资产目录；卸载集群时需要此目录。

10.3.7. 为 RHV 设置 CA 证书

从 Red Hat Virtualization (RHV) Manager 下载 CA 证书，并在安装机器中进行设置。

您可以使用 RHV Manager 的网页或使用 **curl** 命令下载该证书。

之后，您向安装程序提供证书。

流程

1. 使用这两个方法之一下载 CA 证书：

   - 进入 Manager 的网页 **https://<engine-fqdn>/ovirt-engine/**。然后在 下载 中点击 **CA 证书** 链接。
   - 运行以下命令：

     ```
     $ curl -k 'https://<engine-fqdn>/ovirt-engine/services/pki-resource?resource=ca-certificate&format=X509-PEM-CA' -o /tmp/ca.pem
     ```
     (1)

     (1) 对于 **<engine-fqdn>**，请指定 RHV Manager 的全限定域名，如 **rhv-env.virtlab.example.com**。

2. 配置 CA 文件，为 Manager 授予无根用户访问权限。将 CA 文件权限设置为 **0644** （symbolic 值： **-rw-r—r--**）：

   ```
   $ sudo chmod 0644 /tmp/ca.pem
   ```

3. 对于 Linux，将 CA 证书复制到服务器证书目录中。使用 **-p** 保留权限：

   ```
   $ sudo cp -p /tmp/ca.pem /etc/pki/ca-trust/source/anchors/ca.pem
   ```

4. 将证书添加到您操作系统的证书管理器：

   - 对于 macOS，请双击这个证书文件，并使用 **Keychain Access** 程序将该文件添加到 **System** 密钥链中。
   - 对于 Linux，更新 CA 信任：

     ```
     $ sudo update-ca-trust
     ```

注意

如果使用您自己的证书认证机构，请确定系统信任它。

其他资源

- 如需了解更多相关信息，请参阅 RHV 文档中的 身份验证及安全性。

10.3.8. 生成 SSH 私钥并将其添加到代理中

如果要在集群上执行安装调试或灾难恢复，则必须为 **ssh-agent** 和安装程序提供 SSH 密钥。您可以使用此密钥访问公共集群中的 bootstrap 机器来排除安装问题。

注意

在生产环境中，您需要进行灾难恢复和调试。

您可以使用此密钥以 **core** 用户身份通过 SSH 连接到 master 节点。在部署集群时，此密钥会添加到 **core** 用户的 **~/.ssh/authorized_keys** 列表中。

注意

您必须使用一个本地密钥，而不要使用在特定平台上配置的密钥，如 AWS 密钥对。

流程

1. 如果还没有为计算机上免密码身份验证而配置的 SSH 密钥，请创建一个。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ ssh-keygen -t ed25519 -N '' \
  -f <path>/<file_name> 1
```

1 指定新 SSH 密钥的路径和文件名，如 **~/.ssh/id_rsa**。如果您已有密钥对，请确保您的公钥位于 **~/.ssh** 目录中。

运行此命令会在指定的位置生成不需要密码的 SSH 密钥。

注意

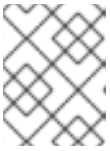
如果您计划在 **x86_64** 架构中安装使用 FIPS 验证的/Modules in Process 加密库的 OpenShift Container Platform 集群，不要创建使用 **ed25519** 算法的密钥。反之，创建一个使用 **rsa** 或 **ecdsa** 算法的密钥。

2. 作为后台任务启动 **ssh-agent** 进程：

```
$ eval "$(ssh-agent -s)"
```

输出示例

```
Agent pid 31874
```



注意

如果您的集群采用 FIPS 模式，则只使用 FIPS 兼容算法来生成 SSH 密钥。密钥必须是 RSA 或 ECDSA。

3. 将 SSH 私钥添加到 **ssh-agent**：

```
$ ssh-add <path>/<file_name> 1
```

输出示例

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

1 指定 SSH 私钥的路径和文件名，如 **~/.ssh/id_rsa**

后续步骤

- 在安装 OpenShift Container Platform 时，为安装程序提供 SSH 公钥。

10.3.9. 获取安装程序

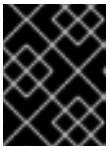
在安装 OpenShift Container Platform 之前，将安装文件下载到本地计算机上。

先决条件

- 运行 Linux 或 macOS 的计算机，本地磁盘空间为 500 MB

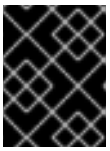
流程

1. 访问 OpenShift Cluster Manager 站点的 Infrastructure Provider 页面。如果您有红帽帐号，请使用自己的凭证登录。如果没有，请创建一个帐户。
2. 选择您的基础架构供应商。
3. 进入适用于您的安装类型的页面，下载您的操作系统的安装程序，并将文件放在要保存安装配置文件的目录中。。



重要

安装程序会在用来安装集群的计算机上创建若干文件。在完成集群安装后，您必须保留安装程序和安装程序所创建的文件。这两个文件都需要删除集群。



重要

删除安装程序创建的文件不会删除您的集群，即使集群在安装过程中失败也是如此。要删除集群，为特定云供应商完成 OpenShift Container Platform 卸载流程。

4. 提取安装程序。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ tar xvf openshift-install-linux.tar.gz
```

5. 从 Red Hat OpenShift Cluster Manager 下载安装 pull secret 。通过此 pull secret，您可以进行所含授权机构提供的服务的身份验证，这些服务包括为 OpenShift Container Platform 组件提供容器镜像的 Quay.io。

10.3.10. 下载 Ansible playbook

下载 Ansible playbook 以在 RHV 上安装 OpenShift Container Platform 版本 4.6。

流程

- 在您的安装机器中运行以下命令：

```
$ mkdir playbooks
```

```
$ cd playbooks
```

```
$ curl -s -L -X GET https://api.github.com/repos/openshift/installer/contents/upi/ovirt?
ref=release-4.6 |
grep 'download_url.*\.yml' |
awk '{ print $2 }' | sed -r 's/("|",)//g' |
xargs -n 1 curl -O
```

-

后续步骤

- 下载这些 Ansible playbook 后，还必须为资产目录创建环境变量，并在运行安装程序创建安装配置文件前自定义 **inventory.yml** 文件。

10.3.11. inventory.yml 文件

您可以使用 **inventory.yml** 文件来定义并创建您要安装的 OpenShift Container Platform 集群的元素。这包括 Red Hat Enterprise Linux CoreOS（RHCOS）镜像、虚拟机模板、bootstrap 机器、control plane 节点和 worker 节点等元素。您还可以使用 **inventory.yml** 来销毁集群。

以下 **inventory.yml** 示例显示参数及其默认值。这些默认值中的数量和数字满足在 RHV 环境中运行生产 OpenShift Container Platform 集群的要求。

inventory.yml 文件示例

```
---
all:
  vars:

    ovirt_cluster: "Default"
    ocp:
      assets_dir: "{{ lookup('env', 'ASSETS_DIR') }}"
      ovirt_config_path: "{{ lookup('env', 'HOME') }}/.ovirt/ovirt-config.yaml"

    # ---
    # {op-system} section
    # ---
    rhcos:
      image_url: "https://mirror.openshift.com/pub/openshift-v4/dependencies/rhcos/4.6/latest/rhcos-openstack.x86_64.qcow2.gz"
      local_cmp_image_path: "/tmp/rhcos.qcow2.gz"
      local_image_path: "/tmp/rhcos.qcow2"

    # ---
    # Profiles section
    # ---
    control_plane:
      cluster: "{{ ovirt_cluster }}"
      memory: 16GiB
      sockets: 4
      cores: 1
      template: rhcos_tpl
      operating_system: "rhcos_x64"
      type: high_performance
      graphical_console:
        headless_mode: false
        protocol:
        - spice
        - vnc
      disks:
      - size: 120GiB
        name: os
        interface: virtio_scsi
```

```
    storage_domain: depot_nvme
  nics:
  - name: nic1
    network: lab
    profile: lab

compute:
  cluster: "{{ ovirt_cluster }}"
  memory: 16GiB
  sockets: 4
  cores: 1
  template: worker_rhcos_tpl
  operating_system: "rhcos_x64"
  type: high_performance
  graphical_console:
    headless_mode: false
    protocol:
    - spice
    - vnc
  disks:
  - size: 120GiB
    name: os
    interface: virtio_scsi
    storage_domain: depot_nvme
  nics:
  - name: nic1
    network: lab
    profile: lab

# ---
# Virtual machines section
# ---
vms:
- name: "{{ metadata.infraID }}-bootstrap"
  ocp_type: bootstrap
  profile: "{{ control_plane }}"
  type: server
- name: "{{ metadata.infraID }}-master0"
  ocp_type: master
  profile: "{{ control_plane }}"
- name: "{{ metadata.infraID }}-master1"
  ocp_type: master
  profile: "{{ control_plane }}"
- name: "{{ metadata.infraID }}-master2"
  ocp_type: master
  profile: "{{ control_plane }}"
- name: "{{ metadata.infraID }}-worker0"
  ocp_type: worker
  profile: "{{ compute }}"
- name: "{{ metadata.infraID }}-worker1"
  ocp_type: worker
  profile: "{{ compute }}"
- name: "{{ metadata.infraID }}-worker2"
  ocp_type: worker
  profile: "{{ compute }}"
```

重要

为描述以 "Enter" 开头的参数输入值。 否则，您可以使用默认值或将其替换为新值。

常规部分

- **ovirt_cluster**：输入现有 RHV 集群的名称，在其中安装 OpenShift Container Platform 集群。
- **OCP.assets_dir**：**openshift-install** 安装程序创建的目录的路径以存储它生成的文件。
- **OCP.ovirt_config_path**：安装程序生成的 **ovirt-config.yaml** 文件的路径，如 **./wrk/install-config.yaml**。此文件包含与管理器的 REST API 交互所需的凭据。

Red Hat Enterprise Linux CoreOS（RHCOS）部分

- **image_url**：输入您指定的用于下载的 RHCOS 镜像的 URL。
- **local_cmp_image_path**：压缩的 RHCOS 镜像的本地下载目录的路径。
- **local_image_path**：提取的 RHCOS 镜像的本地目录路径。

配置集部分

本节由两个配置集组成：

- **control_plane**：bootstrap 和 control plane 节点的配置集。
- **compute**：compute plane 中 worker 节点的配置集。

这些配置集有以下参数。参数的默认值满足运行生产集群的最低要求。您可以增加或自定义这些值来满足您的工作负载要求。

- **Cluster**：这个值从 General Section 中的 **ovirt_cluster** 获取集群名称。
- **memory**：虚拟机的内存量（以 GB 为单位）。
- **socket**：虚拟机的插槽数。
- **cores**：虚拟机的内核数。
- **template**：虚拟机模板的名称。如果计划安装多个集群，且这些集群使用包含不同规格的模板，则使用集群 ID 前附加模板名称。
- **operating_system**：虚拟机中客户端操作系统的类型。对于 oVirt/RHV 版本 4.4，此值必须是 **rhcos_x64**，以便 **Ignition 脚本**的值可以传递给虚拟机。
- **type**：输入 **server** 作为虚拟机的类型。

重要

您必须将 **type** 参数的值从 **high_performance** 改为 **server**。

- **disks**：磁盘规格。**control_plane** 和 **compute** 节点可以有不同的存储域。
- **size**：最小磁盘大小。
- **name**：输入连接到 RHV 中目标集群的磁盘名称。

- **interface**：输入您指定的磁盘接口类型。
- **storage_domain**：输入您指定的磁盘的存储域。
- **nics**：输入虚拟机使用的**名称**和**网络**。您还可以指定虚拟网络接口配置集。默认情况下，NIC 从 oVirt/RHV MAC 池中获取其 MAC 地址。

虚拟机部分

最后部分 **vms** 定义您要在集群中创建和部署的虚拟机。默认情况下，它为生产环境提供最少的 control plane 和 worker 节点数量。

虚拟机包含三个所需的元素：

- **name**：虚拟机的名称。在这种情况下, **metadata.infraID** 会使用 **metadata.yml** 文件中的基础架构 ID 预先填充虚拟机名称。
- **ocp_type**：OCP 集群中的虚拟机的角色。可能的值有 **bootstrap**、**master** 和 **worker**。
- **profile**：每个虚拟机从中继承规格的配置集名称。本例中可能的值是 **control_plane** 或 **compute**。
  您可以覆盖虚拟机从其配置集中继承的值。要做到这一点，您要将配置集属性的名称添加到 **inventory.yml** 中的虚拟机，并为它分配一个覆盖值。要查看这个示例，请检查前面的 **inventory .yml** 示例中的 **name: "{{ metadata.infraID }}-bootstrap"** 虚拟机：它有一个 **type** 属性，其值为 **server**，这会覆盖虚拟机从**control_plane** 配置集继承的 **type** 属性的值。

元数据变量

对于虚拟机，**metadata.infraID** 会利用构建 Ignition 文件时创建的 **metadata.json** 文件中的基础架构 ID 来附加虚拟机的名称。

playbook 使用以下代码从 **ocp.assets_dir** 的特定文件中读取 **infraID**。

```
---
- name: include metadata.json vars
  include_vars:
    file: "{{ ocp.assets_dir }}/metadata.json"
    name: metadata


  ...
```

10.3.12. 指定 RHCOS 镜像设置

更新 **inventory.yml** 文件的 Red Hat Enterprise Linux CoreOS（RHCOS）镜像设置。之后，当您运行此文件之一时，它会将压缩的 Red Hat Enterprise Linux CoreOS（RHCOS）镜像从 **image_url** URL 下载到 **local_cmp_image_path** 目录。然后，playbook 会将镜像解压缩到 **local_image_path** 目录，并使用它来创建 oVirt/RHV 模板。

流程

1. 找到您要安装的 OpenShift Container Platform 版本的 RHCOS 镜像下载页面，如 /pub/openshift-v4/dependencies/rhcos/latest/latest/latest/latest/latest。
2. 在该下载页面中复制 OpenStack **qcow2** 镜像的 URL，如 **https://mirror.openshift.com/pub/openshift-v4/dependencies/rhcos/4.6/latest/rhcos-openstack.x86_64.qcow2.gz**。

3. 编辑之前下载的 **inventory.yml** playbook。此时会粘贴 URL 作为 **image_url** 的值。例如：

```
rhcos:
  "https://mirror.openshift.com/pub/openshift-v4/dependencies/rhcos/4.6/latest/rhcos-openstack.x86_64.qcow2.gz"
```

10.3.13. 创建安装配置文件

您可以通过运行安装程序（**openshift-install**）并使用之前指定或收集的信息响应其提示来创建安装配置文件。

当完成对提示的响应后，安装程序会在之前指定的 asset 目录中创建 **install-config.yaml** 文件的初始版本，如 **./wrk/install-config.yaml**

安装程序还会创建一个文件 **$HOME/.ovirt/ovirt-config.yaml**，其中包含访问 Manager 并使用其 REST API 所需的所有连接参数。

注： 安装过程不使用您为一些参数提供的值，如**内部 API 虚拟 IP** 和 **Ingress 虚拟 IP**，因为您已在基础架构 DNS 中配置了这些参数。

它还使用您在 **inventory.yml** 中为参数提供的值，如 **oVirt cluster**、**oVirt storage** 和 **oVirt network**。使用一个脚本删除或替换 **install-config.yaml** 中的相同值，使用前面提到的**虚拟 IP**。

流程

1. 运行安装程序：

```
$ openshift-install create install-config --dir $ASSETS_DIR
```

2. 根据安装程序的提示输入您系统的信息。

输出示例

```
? SSH Public Key /home/user/.ssh/id_dsa.pub
? Platform <ovirt>
? Engine FQDN[:PORT] [? for help] <engine.fqdn>
? Enter ovirt-engine username <ocpadmin@internal>
? Enter password <******>
? oVirt cluster <cluster>
? oVirt storage <storage>
? oVirt network <net>
? Internal API virtual IP <172.16.0.252>
? Ingress virtual IP <172.16.0.251>
? Base Domain <example.org>
? Cluster Name <ocp4>
? Pull Secret [? for help] <********>
```

```
? SSH Public Key /home/user/.ssh/id_dsa.pub
? Platform <ovirt>
? Engine FQDN[:PORT] [? for help] <engine.fqdn>
? Enter ovirt-engine username <ocpadmin@internal>
? Enter password <******>
? oVirt cluster <cluster>
? oVirt storage <storage>
```

```
? oVirt network <net>
? Internal API virtual IP <172.16.0.252>
? Ingress virtual IP <172.16.0.251>
? Base Domain <example.org>
? Cluster Name <ocp4>
? Pull Secret [? for help] <********>
```

对于 **Internal API 虚拟 IP** 和 **Ingress 虚拟 IP**，请提供您在配置 DNS 服务时指定的 IP 地址。

您输入 **oVirt 集群** 和 **Base Domain** 的值一起形成 REST API 的 URL 的 FQDN 部分以及您创建的所有应用程序，如 **https://api.ocp4.example.org:6443/** 和 **https://console-openshift-console.apps.ocp4.example.org**。

您可以从 Red Hat OpenShift Cluster Manager 获取 pull secret 。

10.3.14. 自定义 install-config.yaml

在这里，您使用三个 Python 脚本覆盖一些安装程序的默认行为：

- 默认情况下，安装程序使用机器 API 创建节点。要覆盖此默认行为，将计算节点数量设置为零个副本。之后，您可以使用 Ansible playbook 创建计算节点。
- 默认情况下，安装程序为节点设置机器网络的 IP 范围。要覆盖这个默认行为，您可以将 IP 范围设置为与您的基础架构匹配。
- 默认情况下，安装程序将平台设置为 **ovirt**。但是，在用户置备的基础架构上安装集群和在裸机上安装集群更为相似。因此，您可以从 **install-config.yaml** 中删除 ovirt platform 部分，并将平台改为 **none**。然后，使用 **inventory.yml** 指定所有所需的设置。

注意

这些片断可用于 Python 3 和 Python 2。

流程

1. 将计算节点数量设置为零副本：

```
$ python3 -c 'import os, yaml
path = "%s/install-config.yaml" % os.environ["ASSETS_DIR"]
conf = yaml.safe_load(open(path))
conf["compute"][0]["replicas"] = 0
open(path, "w").write(yaml.dump(conf, default_flow_style=False))'
```

2. 设置机器网络的 IP 范围。例如，要将范围设置为 **172.16.0.0/16**，请输入：

```
$ python3 -c 'import os, yaml
path = "%s/install-config.yaml" % os.environ["ASSETS_DIR"]
conf = yaml.safe_load(open(path))
conf["networking"]["machineNetwork"][0]["cidr"] = "172.16.0.0/16"
open(path, "w").write(yaml.dump(conf, default_flow_style=False))'
```

3. 删除 **ovirt** 部分，把平台改为 **none**:

```
$ python3 -c 'import os, yaml
```

```
path = "%s/install-config.yaml" % os.environ["ASSETS_DIR"]
conf = yaml.safe_load(open(path))
platform = conf["platform"]
del platform["ovirt"]
platform["none"] = {}
open(path, "w").write(yaml.dump(conf, default_flow_style=False))'
```

10.3.15. 生成清单文件

使用安装程序在 asset 目录中生成一组清单文件。

生成清单文件的命令在消耗 **install-config.yaml** 文件前会显示警告消息。

如果您计划重复使用 **install-config.yaml** 文件，请在生成清单文件前生成备份副本。

流程

1. 可选：创建 **install-config.yaml** 文件的备份副本：

```
$ cp install-config.yaml install-config.yaml.backup
```

2. 在资产目录中生成一组清单：

```
$ openshift-install create manifests --dir $ASSETS_DIR
```

该命令显示以下信息。

输出示例

```
INFO Consuming Install Config from target directory
WARNING Making control-plane schedulable by setting MastersSchedulable to true for
Scheduler cluster settings
```

该命令生成以下清单文件：

输出示例

```
$ tree
.
└── wrk
    ├── manifests
    │   ├── 04-openshift-machine-config-operator.yaml
    │   ├── cluster-config.yaml
    │   ├── cluster-dns-02-config.yml
    │   ├── cluster-infrastructure-02-config.yml
    │   ├── cluster-ingress-02-config.yml
    │   ├── cluster-network-01-crd.yml
    │   ├── cluster-network-02-config.yml
    │   ├── cluster-proxy-01-config.yaml
    │   ├── cluster-scheduler-02-config.yml
    │   ├── cvo-overrides.yaml
    │   ├── etcd-ca-bundle-configmap.yaml
    │   ├── etcd-client-secret.yaml
    │   ├── etcd-host-service-endpoints.yaml
```

```
│   ├── etcd-host-service.yaml
│   ├── etcd-metric-client-secret.yaml
│   ├── etcd-metric-serving-ca-configmap.yaml
│   ├── etcd-metric-signer-secret.yaml
│   ├── etcd-namespace.yaml
│   ├── etcd-service.yaml
│   ├── etcd-serving-ca-configmap.yaml
│   ├── etcd-signer-secret.yaml
│   ├── kube-cloud-config.yaml
│   ├── kube-system-configmap-root-ca.yaml
│   ├── machine-config-server-tls-secret.yaml
│   └── openshift-config-secret-pull-secret.yaml
└── openshift
    ├── 99_kubeadmin-password-secret.yaml
    ├── 99_openshift-cluster-api_master-user-data-secret.yaml
    ├── 99_openshift-cluster-api_worker-user-data-secret.yaml
    ├── 99_openshift-machineconfig_99-master-ssh.yaml
    ├── 99_openshift-machineconfig_99-worker-ssh.yaml
    └── openshift-install-manifests.yaml
```

后续步骤

- 使 control-plane 节点不可调度。

10.3.16. 使 control-plane 节点不可调度

由于要手动创建和部署 control plane 机器，所以您必须配置清单文件，使 control-plane 节点不可调度。

流程

1. 要使 control-plane 节点不可调度，请输入：

```
$ python3 -c 'import os, yaml
path = "%s/manifests/cluster-scheduler-02-config.yml" % os.environ["ASSETS_DIR"]
data = yaml.safe_load(open(path))
data["spec"]["mastersSchedulable"] = False
open(path, "w").write(yaml.dump(data, default_flow_style=False))'
```

10.3.17. 构建 Ignition 文件

要从您刚才生成和修改的清单文件构建 Ignition 文件，请运行安装程序。此操作会创建一个 Red Hat Enterprise Linux CoreOS（RHCOS）机器 **initramfs**，它将获取 Ignition 文件并执行创建节点所需的配置。

除了 Ignition 文件外，安装程序还会生成以下内容：

- 包含使用 **oc** 和 **kubectl** 工具连接到集群的 admin 凭证的 **auth** 目录。
- 包含当前安装的 OpenShift Container Platform 集群名称、集群 ID 和基础架构 ID 的 **metadata.json** 文件。

此安装过程的 Ansible playbook 使用 **infraID** 值作为它们创建的虚拟机的前缀。这可防止在同一 oVirt/RHV 集群中有多个安装时的命名冲突。

注意

Ignition 配置文件中的证书会在 24 小时后过期。完成集群安装，并将集群以非降级状态持续运行 24 小时，以便完成第一次证书轮转。

流程

1. 要构建 Ignition 文件，请输入：

```
$ openshift-install create ignition-configs --dir $ASSETS_DIR
```

输出示例

```
$ tree
.
└── wrk
    ├── auth
    │   ├── kubeadmin-password
    │   └── kubeconfig
    ├── bootstrap.ign
    ├── master.ign
    ├── metadata.json
    └── worker.ign
```

10.3.18. 创建模板和虚拟机

在确认 **inventory.yml** 中的变量后，您要运行第一个 Ansible 置备 playbook **create-templates-and-vms.yml**。

此 playbook 使用 **$HOME/.ovirt/ovirt-config.yaml** 中的 RHV Manager 的连接参数，并在资产目录中读取 **metadata.json**。

如果本地 Red Hat Enterprise Linux CoreOS（RHCOS）镜像不存在，则 playbook 会从您为 **inventory.yml** 中的 **image_url** 指定的 URL 下载一个。它提取镜像并将其上传到 RHV 以创建模板。

playbook 根据 **inventory.yml** 文件中的 **control_plane** 和 **compute** 配置集创建一个模板。如果这些配置集有不同的名称，它会创建两个模板。

playbook 完成后，其创建的虚拟机将停止。您可以从中获取信息来帮助配置其他基础架构元素。例如，您可以获取虚拟机的 MAC 地址来配置 DHCP，为虚拟机分配永久 IP 地址。

流程

1. 在 **inventory.yml** 中，在 **control_plane** 和 **compute** 变量下，将 **type: high_performance** 的两个实例更改为 **type: server**。
2. 可选：如果您计划在同一集群上执行多个安装，请为每个 OCP 安装创建不同的模板。在 **inventory.yml** 文件中，使用 **infraID** 预先填充 **模板** 值。例如：

```
control_plane:
  cluster: "{{ ovirt_cluster }}"
  memory: 16GiB
  sockets: 4
  cores: 1
```

```
    template: "{{ metadata.infraID }}-rhcos_tpl"
    operating_system: "rhcos_x64"
    ...
```

3. 创建模板和虚拟机：

```
$ ansible-playbook -i inventory.yml create-templates-and-vms.yml
```

10.3.19. 创建 bootstrap 机器

您可以通过运行 **bootstrap.yml** playbook 创建 bootstrap 机器。此 playbook 启动 bootstrap 虚拟机，并从 asset 目录中传递 **bootstrap.ign** Ignition 文件。bootstrap 节点配置自己，使其能为 control plane 节点提供 Ignition 文件。

要监控 bootstrap 过程，您可以使用 RHV 管理门户中的控制台或使用 SSH 连接到虚拟机。

流程

1. 创建 bootstrap 机器：

```
$ ansible-playbook -i inventory.yml bootstrap.yml
```

2. 使用管理入口或 SSH 中的控制台连接到 bootstrap 机器。将 **<bootstrap_ip>** 替换为 bootstrap 节点的 IP 地址。要使用 SSH，请输入：

```
$ ssh core@<boostrap.ip>
```

3. 从 bootstrap 节点收集发行镜像服务的 **bootkube.service** journald 单元日志：

```
[core@ocp4-lk6b4-bootstrap ~]$ journalctl -b -f -u release-image.service -u bootkube.service
```

注意

bootstrap 节点上的 **bootkube.service** 日志会输出 etcd **connection refused** 错误，这表示 bootstrap 服务器无法在 control plane 节点（也称为 master 节点）上连接到 etcd。在各个 control plane 节点上启动 etcd 且节点已加入集群后，这个错误应该会停止。

10.3.20. 创建 control plane 节点

您可以通过运行 **masters.yml** playbook 来创建 control plane 节点。此 playbook 将 **master.ign** Ignition 文件传递给每个虚拟机。Ignition 文件包含 control plane 节点的指令，可从 URL（如 **https://api-int.ocp4.example.org:22623/config/master**）获取 Ignition。这个 URL 中的端口号由负载均衡器管理，且只能在集群中访问。

流程

1. 创建 control plane 节点：

```
$ ansible-playbook -i inventory.yml masters.yml
```

2. 在 playbook 创建 control plane 时，监控 bootstrap 过程：

```
$ openshift-install wait-for bootstrap-complete --dir $ASSETS_DIR
```

输出示例

```
INFO API v1.18.3+b74c5ed up
INFO Waiting up to 40m0s for bootstrapping to complete...
```

3. 当 control plane 节点上所有 pod 都启动并运行 etcd 时，安装程序会显示以下输出。

输出示例

```
INFO It is now safe to remove the bootstrap resources
```

10.3.21. 验证集群状态

您可以在安装过程中或安装后验证 OpenShift Container Platform 集群的状态：

流程

1. 在集群环境中，导出管理员的 kubeconfig 文件：

```
$ export KUBECONFIG=$ASSETS_DIR/auth/kubeconfig
```

kubeconfig 文件包含关于集群的信息，供 CLI 用于将客户端连接到正确集群和 API 服务器。

2. 查看部署后创建的 control plane 和计算机器：

```
$ oc get nodes
```

3. 查看集群的版本：

```
$ oc get clusterversion
```

4. 查看 Operator 的状态：

```
$ oc get clusteroperator
```

5. 查看集群中的所有正在运行的 pod:

```
$ oc get pods -A
```

10.3.22. 删除 bootstrap 机器

在 **wait-for** 命令显示 bootstrap 过程完成后，您必须删除 bootstrap 虚拟机来释放计算、内存和存储资源。另外，从负载均衡器指令中删除 bootstrap 机器的设置。

流程

1. 要从集群中删除 bootstrap 机器，请输入：

```
$ ansible-playbook -i inventory.yml retire-bootstrap.yml
```

2. 从负载均衡器指令中删除 bootstrap 机器的设置。

10.3.23. 创建 worker 节点并完成安装

创建 worker 节点与创建 control plane 节点类似。但是，worker 节点 worker 不会自动加入集群。要将其添加到集群中，请检查并批准 worker 的待处理的 CSR（Certificate Signing Requests）。

批准第一个请求后，您将继续批准 CSR，直到所有 worker 节点都被批准为止。完成此过程后，worker 节点就变为 **Ready**，并且可以调度在其上运行的 pod。

最后，使用命令行查看安装过程何时完成。

流程

1. 创建 worker 节点：

```
$ ansible-playbook -i inventory.yml workers.yml
```

2. 要列出所有 CSR，请输入：

```
$ oc get csr -A
```

最后，这个命令会显示每个节点的一个 CSR。例如：

输出示例

```
NAME        AGE    SIGNERNAME                                    REQUESTOR
CONDITION
csr-2lnxd   63m    kubernetes.io/kubelet-serving                 system:node:ocp4-lk6b4-
master0.ocp4.example.org                                         Approved,Issued
csr-hff4q   64m    kubernetes.io/kube-apiserver-client-kubelet
system:serviceaccount:openshift-machine-config-operator:node-bootstrapper
Approved,Issued
csr-hsn96   60m    kubernetes.io/kubelet-serving                 system:node:ocp4-lk6b4-
master2.ocp4.example.org                                         Approved,Issued
csr-m724n   6m2s   kubernetes.io/kube-apiserver-client-kubelet
system:serviceaccount:openshift-machine-config-operator:node-bootstrapper   Pending
csr-p4dz2   60m    kubernetes.io/kube-apiserver-client-kubelet
system:serviceaccount:openshift-machine-config-operator:node-bootstrapper
Approved,Issued
csr-t9vfj   60m    kubernetes.io/kubelet-serving                 system:node:ocp4-lk6b4-
master1.ocp4.example.org                                         Approved,Issued
csr-tggtr   61m    kubernetes.io/kube-apiserver-client-kubelet
system:serviceaccount:openshift-machine-config-operator:node-bootstrapper
Approved,Issued
csr-wcbrf   7m6s   kubernetes.io/kube-apiserver-client-kubelet
system:serviceaccount:openshift-machine-config-operator:node-bootstrapper   Pending
```

3. 要过滤列表并只查看待处理的 CSR，请输入：

```
$ watch "oc get csr -A | grep pending -i"
```

此命令每两秒钟刷新输出一次，仅显示待处理的 CSR。例如：

输出示例

```
Every 2.0s: oc get csr -A | grep pending -i

csr-m724n   10m   kubernetes.io/kube-apiserver-client-kubelet
system:serviceaccount:openshift-machine-config-operator:node-bootstrapper   Pending
csr-wcbrf   11m   kubernetes.io/kube-apiserver-client-kubelet
system:serviceaccount:openshift-machine-config-operator:node-bootstrapper   Pending
```

4. 检查每个待处理的请求。例如：

输出示例

```
$ oc describe csr csr-m724n
```

输出示例

```
Name:               csr-m724n
Labels:             <none>
Annotations:        <none>
CreationTimestamp:  Sun, 19 Jul 2020 15:59:37 +0200
Requesting User:    system:serviceaccount:openshift-machine-config-operator:node-
bootstrapper
Signer:             kubernetes.io/kube-apiserver-client-kubelet
Status:             Pending
Subject:
        Common Name:    system:node:ocp4-lk6b4-worker1.ocp4.example.org
        Serial Number:
        Organization:   system:nodes
Events:  <none>
```

5. 如果 CSR 信息正确，则批准请求：

```
$ oc adm certificate approve csr-m724n
```

6. 等待安装过程完成：

```
$ openshift-install wait-for install-complete --dir $ASSETS_DIR --log-level debug
```

安装完成后，命令行会显示 OpenShift Container Platform Web 控制台以及管理员用户名和密码的 URL。

10.3.24. OpenShift Container Platform 的 Telemetry 访问

在 OpenShift Container Platform 4.6 中，默认运行的 Telemetry 服务提供有关集群健康状况和成功更新的指标，需要访问互联网。如果您的集群连接到互联网，Telemetry 会自动运行，而且集群会注册到 OpenShift Cluster Manager。

确认 OpenShift Cluster Manager 清单正确后，可以由 Telemetry 自动维护，也可以使用 OpenShift Cluster Manager 手动维护，使用订阅监控来跟踪帐户或多集群级别的 OpenShift Container Platform 订阅。

其他资源

- 有关 Telemetry 服务的更多信息，请参阅关于远程健康监控。

10.4. 在 RHV 上卸载集群

您可以从 Red Hat Virtualization (RHV) 中删除 OpenShift Container Platform 集群。

10.4.1. 删除使用安装程序置备的基础架构的集群

您可以从云中删除使用安装程序置备的基础架构的集群。

注意

卸载后，检查云供应商是否有没有被正确移除的资源，特别是 User Provisioned Infrastructure（UPI）集群。可能存在安装程序没有创建的资源，或者安装程序无法访问的资源。

先决条件

- 有部署集群时所用的安装程序副本。
- 有创建集群时安装程序所生成的文件。

流程

1. 在用来安装集群的计算机中包含安装程序的目录中，运行以下命令：

```
$ ./openshift-install destroy cluster \
--dir <installation_directory> --log-level info 1 2
```

1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

2 要查看不同的详情，请指定 **warn**、**debug** 或 **error**，而不要指定 **info**。

注意

您必须为集群指定包含集群定义文件的目录。安装程序需要此目录中的 **metadata.json** 文件来删除集群。

2. 可选：删除 **<installation_directory>** 目录和 OpenShift Container Platform 安装程序。

10.4.2. 删除使用用户置备的基础架构的集群

在使用完集群后,您可以从云中删除使用用户置备的基础架构的集群。

先决条件

- 具有用于安装集群的原始 playbook 文件、资产目录文件以及 **$ASSETS_DIR** 环境变量。通常，您可以通过使用安装集群时所用的同一计算机来达到此目的。

流程

1. 要删除集群，请输入：

```
$ ansible-playbook -i inventory.yml \
    retire-bootstrap.yml \
    retire-masters.yml   \
    retire-workers.yml
```

2. 删除您添加到 DNS、负载均衡器以及此集群的任何其他基础架构的任何配置。

第 11 章 在 VSPHERE 上安装

11.1. 在 VSPHERE 上安装集群

在 OpenShift Container Platform 版本 4.6 中，您可以使用安装程序置备的基础架构在 VMware vSphere 实例上安装集群。

注意

OpenShift Container Platform 支持将集群部署到单个 VMware vCenter 中。不支持在多个 vCenter 上使用机器/机器集部署集群。

11.1.1. 先决条件

- 为集群置备持久性存储。若要部署私有镜像 registry，您的存储必须提供 **ReadWriteMany** 访问模式。
- 查看有关 OpenShift Container Platform 安装和更新流程的详细信息。
- OpenShift Container Platform 安装程序需要访问 vCenter 和 ESXi 主机上的端口 443。您确认可以访问端口 443。
- 如果您使用防火墙，您与管理员确认可以访问端口 443。control plane 节点必须能够通过端口 443 访问 vCenter 和 ESXi 主机，才能成功安装。
- 如果使用防火墙，则必须将其配置为允许集群需要访问的站点。

注意

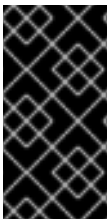
如果您要配置代理，请务必也要查看此站点列表。

11.1.2. OpenShift Container Platform 的互联网访问

在 OpenShift Container Platform 4.6 中，您需要访问互联网来安装集群。

您必须具有以下互联网访问权限：

- 访问 OpenShift Cluster Manager 以下载安装程序并执行订阅管理。如果集群可以访问互联网，并且没有禁用 Telemetry，该服务会自动授权您的集群。
- 访问 Quay.io，以获取安装集群所需的软件包。
- 获取执行集群更新所需的软件包。



重要

如果您的集群无法直接访问互联网，则可以在置备的某些类基础架构上执行受限网络安装。在此过程中，您要下载所需的内容，并使用它在镜像 registry（mirror registry） 中填充安装集群并生成安装程序所需的软件包。对于某些安装类型，集群要安装到的环境不需要访问互联网。在更新集群之前，要更新 registry 镜像系统中的内容。

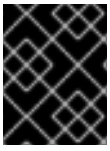
11.1.3. VMware vSphere 基础架构要求

您必须在满足您使用的组件要求的 VMware vSphere 版本 6 或 7 实例上安装 OpenShift Container Platform 集群。

表 11.1. VMware 组件支持的最低 vSphere 版本

组件	最低支持版本	描述
虚拟机监控程序	vSphere 6.5 及之后的版本 13	此版本是 Red Hat Enterprise Linux CoreOS(RHCOS)支持的最低版本。请查看 Red Hat Enterprise Linux 8 支持的管理程序列表。
使用 in-tree 驱动程序存储	vSphere 6.5 及之后的版本	此插件使用 OpenShift Container Platform 中包含的 vSphere 的树内存储驱动程序创建 vSphere 存储。
可选： Networking (NSX-T)	vSphere 6.5U3 或 vSphere 6.7U2 及之后的版本	OpenShift Container Platform 需要 vSphere 6.5U3 或 vSphere 6.7U2+。VMware 的 NSX Container Plug-in（NCP）3.0.2 使用 OpenShift Container Platform 4.6 和 NSX-T 3.x+ 认证。

如果您使用 vSphere 版本 6.5 实例，请在安装 OpenShift Container Platform 前考虑升级到 6.7U3 或 7.0。



重要

您必须确保在安装 OpenShift Container Platform 前同步 ESXi 主机上的时间。请参阅 VMware 文档中的编辑主机时间配置。

11.1.4. 网络连接要求

您必须配置机器之间的网络连接，以允许 OpenShift Container Platform 集群组件进行通信。

查看有关所需网络端口的以下详细信息。

表 11.2. 用于全机器到所有机器通信的端口

协议	端口	描述
ICMP	N/A	网络可访问性测试
TCP	**1936**	指标
	9000-9999	主机级别的服务，包括端口 **9100-9101** 上的节点导出器和端口 **9099** 上的 Cluster Version Operator。

协议	端口	描述
	10250-10259	Kubernetes 保留的默认端口
	10256	openshift-sdn
UDP	**4789**	虚拟可扩展 LAN(VXLAN)
	6081	Geneve
	9000-9999	主机级别的服务，包括端口 **9100-9101** 上的节点导出器。
	500	IPsec IKE 数据包
	4500	IPsec NAT-T 数据包
TCP/UDP	**30000-32767**	Kubernetes 节点端口
ESP	N/A	IPsec Encapsulating Security Payload(ESP)

表 11.3. 用于所有机器控制平面通信的端口

协议	端口	描述
TCP	**6443**	Kubernetes API

表 11.4. control plane 机器用于 control plane 机器通信的端口

协议	端口	描述
TCP	**2379-2380**	etcd 服务器和对等端口

11.1.5. vCenter 要求

在使用安装程序置备的基础架构的 vCenter 上安装 OpenShift Container Platform 集群前，您必须准备自己的环境。

所需的 vCenter 帐户权限
要在 vCenter 中安装 OpenShift Container Platform 集群，安装程序需要一个具有特权的帐户来读取和创建所需资源。使用具有全局管理特权的帐户是访问所有必要权限的最简单方式。

如果无法使用具有全局管理特权的帐户，您必须创建角色来授予 OpenShift Container Platform 集群安装所需的权限。虽然大多数权限始终是必需的，但是一些权限只有在计划安装程序需要在您的 vCenter 实例中置备一个包含 OpenShift Container Platform 集群的文件夹时（这是默认行为）才需要。您必须为指定对象创建或修改 vSphere 角色，才能授予所需的权限。

如果安装程序创建 vSphere 虚拟机文件夹，则需要额外的角色。

例 11.1. 安装所需的角色和权限

角色的 vSphere 对象	何时需要	所需的权限
vSphere vCenter	Always	**Cns.Searchable** **InventoryService.Tagging.AttachTag** **InventoryService.Tagging.CreateCategory** **InventoryService.Tagging.CreateTag** **InventoryService.Tagging.DeleteCategory** **InventoryService.Tagging.DeleteTag** **InventoryService.Tagging.EditCategory** **InventoryService.Tagging.EditTag** **Sessions.ValidateSession** **StorageProfile.View**
vSphere vCenter Cluster	Always	**Host.Config.Storage** **Resource.AssignVMToPool** **VApp.AssignResourcePool** **VApp.Import** **VirtualMachine.Config.AddNewDisk**
vSphere Datastore	Always	**Datastore.AllocateSpace** **Datastore.Browse** **Datastore.FileManagement**
vSphere 端口组	Always	**Network.Assign**

角色的 vSphere 对象	何时需要	所需的权限
虚拟机文件夹	Always	**Resource.AssignVMToPool VApp.Import VirtualMachine.Config.Add ExistingDisk VirtualMachine.Config.Add NewDisk VirtualMachine.Config.Add RemoveDevice VirtualMachine.Config.Adva ncedConfig VirtualMachine.Config.Anno tation VirtualMachine.Config.CPU Count VirtualMachine.Config.Disk Extend VirtualMachine.Config.Disk Lease VirtualMachine.Config.Edit Device VirtualMachine.Config.Mem ory VirtualMachine.Config.Rem oveDisk VirtualMachine.Config.Rena me VirtualMachine.Config.Rese tGuestInfo VirtualMachine.Config.Reso urce VirtualMachine.Config.Setti ngs VirtualMachine.Config.Upgr adeVirtualHardware VirtualMachine.Interact.Gue stControl VirtualMachine.Interact.Pow erOff VirtualMachine.Interact.Pow erOn VirtualMachine.Interact.Res et VirtualMachine.Inventory.Cr eate VirtualMachine.Inventory.Cr eateFromExisting VirtualMachine.Inventory.D elete VirtualMachine.Provisionin g.Clone**

角色的 vSphere 对象	何时需要	所需的权限
vSphere vCenter Datacenter	如果安装程序创建虚拟机文件夹	**Resource.AssignVMToPool** **VApp.Import** **VirtualMachine.Config.AddExistingDisk** **VirtualMachine.Config.AddNewDisk** **VirtualMachine.Config.AddRemoveDevice** **VirtualMachine.Config.AdvancedConfig** **VirtualMachine.Config.Annotation** **VirtualMachine.Config.CPUCount** **VirtualMachine.Config.DiskExtend** **VirtualMachine.Config.DiskLease** **VirtualMachine.Config.EditDevice** **VirtualMachine.Config.Memory** **VirtualMachine.Config.RemoveDisk** **VirtualMachine.Config.Rename** **VirtualMachine.Config.ResetGuestInfo** **VirtualMachine.Config.Resource** **VirtualMachine.Config.Settings** **VirtualMachine.Config.UpgradeVirtualHardware** **VirtualMachine.Interact.GuestControl** **VirtualMachine.Interact.PowerOff** **VirtualMachine.Interact.PowerOn** **VirtualMachine.Interact.Reset** **VirtualMachine.Inventory.Create** **VirtualMachine.Inventory.CreateFromExisting** **VirtualMachine.Inventory.Delete** **VirtualMachine.Provisioning.Clone** **Folder.Create** **Folder.Delete**

此外，用户需要一些 **ReadOnly** 权限，某些角色需要权限来提升对子对象的权限。这些设置会根据您是否将集群安装到现有文件夹而有所不同。

例 11.2. 所需的权限和传播设置

vSphere 对象	文件夹类型	传播到子对象	所需的权限
vSphere vCenter	Always	False	列出所需的权限
vSphere vCenter Datacenter	现有文件夹	False	**ReadOnly** 权限
	安装程序创建文件夹	True	列出所需的权限
vSphere vCenter Cluster	Always	True	列出所需的权限
vSphere vCenter Datastore	Always	False	列出所需的权限
vSphere Switch	Always	False	**ReadOnly** 权限
vSphere 端口组	Always	False	列出所需的权限
vSphere vCenter Virtual Machine Folder	现有文件夹	True	列出所需的权限

有关只使用所需权限创建帐户的更多信息，请参阅 vSphere 文档中的 vSphere 权限和用户管理任务。

将 OpenShift Container Platform 与 vMotion 搭配使用

如果要在 vSphere 环境中使用 vMotion，请在安装 OpenShift Container Platform 集群前考虑以下内容。

- OpenShift Container Platform 通常支持仅用于计算的 vMotion。使用 Storage vMotion 可能会导致问题且不被支持。
  为了帮助确保计算和 control plane 节点的正常运行时间，建议您遵循 VMware 最佳实践进行 vMotion。还建议使用 VMware 反关联性规则来改进 OpenShift Container Platform 在维护或硬件问题期间的可用性。

  有关 vMotion 和 anti-affinity 规则的更多信息，请参阅 VMware vSphere 文档 了解 vMotion 网络要求和虚拟机反关联性规则。

- 如果您在 pod 中使用 vSphere 卷，请手动或通过 Storage vMotion 在数据存储间迁移虚拟机，从而导致 OpenShift Container Platform 持久性卷(PV)对象中的无效引用。这些引用可防止受影响的 pod 启动，并可能导致数据丢失。

- 同样，OpenShift Container Platform 不支持在数据存储间有选择地迁移 VMDK、使用数据存储集群进行虚拟机置备、动态或静态置备 PV，或使用作为数据存储集群一部分的数据存储进行 PV 的动态或静态置备。

集群资源
当部署使用安装程序置备的基础架构的 OpenShift Container Platform 集群时，安装程序必须能够在 vCenter 实例中创建多个资源。

标准 OpenShift Container Platform 安装会创建以下 vCenter 资源：

- 1 个文件夹
- 1 标签（Tag）类别
- 1 个标签（Tag）
- 虚拟机:
  - 1 个模板
  - 1 个临时 bootstrap 节点
  - 3 个 control plane 节点
  - 3 个计算机器

虽然这些资源使用了 856 GB 存储，但 bootstrap 节点会在集群安装过程中被销毁。使用标准集群至少需要 800 GB 存储。

如果部署了更多计算机器，OpenShift Container Platform 集群将使用更多存储。

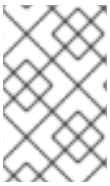
集群的限制
可用资源因集群而异。vCenter 中可能的集群数量主要受可用存储空间以及对所需资源数量的限制。确保考虑集群创建的 vCenter 资源的限制和部署集群所需的资源，如 IP 地址和网络。

网络要求
网络必须使用 DHCP，并确保 DHCP 服务器被配置为为集群机器提供持久的 IP 地址。

注意

在安装开始前，持久性 IP 地址不可用。分配 DHCP 范围后，在安装后使用持久 IP 地址手动替换分配。

另外，在安装 OpenShift Container Platform 集群前，必须创建以下网络资源：



注意

建议集群中的每个 OpenShift Container Platform 节点都可以访问可通过 DHCP 发现的网络时间协议（NTP）服务器。没有 NTP 服务器也可安装。但是，异步服务器时钟将导致错误，NTP 服务器会阻止。

所需的 IP 地址
安装程序置备的 vSphere 安装需要这些静态 IP 地址：

- API 地址用于访问集群 API。
- Ingress 地址用于集群入口流量。
- 当将集群从版本 4.5 升级到 4.6 时，会使用 control plane 节点地址。

安装 OpenShift Container Platform 集群时，必须向安装程序提供这些 IP 地址。

DNS 记录

您必须在正确的 DNS 服务器中为托管 OpenShift Container Platform 集群的 vCenter 实例创建两个静态 IP 地址的 DNS 记录。在每个记录中，**<cluster_name>** 是集群名称，**<base_domain>** 是您在安装集群时指定的集群基域。完整的 DNS 记录采用如下格式: **<component>.<cluster_name>.<base_domain>.**。

表 11.5. 所需的 DNS 记录

组件	记录	描述
API VIP	**api.<cluster_name>.<base_domain>.**	此 DNS A/AAAA 或 CNAME 记录必须指向 control plane 机器的负载均衡器。此记录必须能由集群外的客户端和集群内的所有节点解析。
Ingress VIP	***.apps.<cluster_name>.<base_domain>.**	通配符 DNS A/AAAA 或 CNAME 记录，指向以运行入口路由器 Pod 的机器（默认为 worker 节点）为目标的负载均衡器。此记录必须能由集群外的客户端和集群内的所有节点解析。

11.1.6. 生成 SSH 私钥并将其添加到代理中

如果要在集群上执行安装调试或灾难恢复，则必须为 **ssh-agent** 和安装程序提供 SSH 密钥。您可以使用此密钥访问公共集群中的 bootstrap 机器来排除安装问题。

注意

在生产环境中，您需要进行灾难恢复和调试。

您可以使用此密钥以 **core** 用户身份通过 SSH 连接到 master 节点。在部署集群时，此密钥会添加到 **core** 用户的 **~/.ssh/authorized_keys** 列表中。

注意

您必须使用一个本地密钥，而不要使用在特定平台上配置的密钥，如 AWS 密钥对。

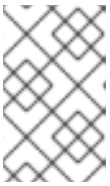
流程

1. 如果还没有为计算机上免密码身份验证而配置的 SSH 密钥，请创建一个。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ ssh-keygen -t ed25519 -N '' \
  -f <path>/<file_name> 1
```

1 指定新 SSH 密钥的路径和文件名，如 **~/.ssh/id_rsa**。如果您已有密钥对，请确保您的公钥位于 **~/.ssh** 目录中。

运行此命令会在指定的位置生成不需要密码的 SSH 密钥。



注意

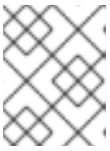
如果您计划在 **x86_64** 架构中安装使用 FIPS 验证的/Modules in Process 加密库的 OpenShift Container Platform 集群，不要创建使用 **ed25519** 算法的密钥。反之，创建一个使用 **rsa** 或 **ecdsa** 算法的密钥。

2. 作为后台任务启动 **ssh-agent** 进程：

```
$ eval "$(ssh-agent -s)"
```

输出示例

```
Agent pid 31874
```



注意

如果您的集群采用 FIPS 模式，则只使用 FIPS 兼容算法来生成 SSH 密钥。密钥必须是 RSA 或 ECDSA。

3. 将 SSH 私钥添加到 **ssh-agent**：

```
$ ssh-add <path>/<file_name> 1
```

输出示例

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

1 指定 SSH 私钥的路径和文件名，如 **~/.ssh/id_rsa**

后续步骤

- 在安装 OpenShift Container Platform 时，为安装程序提供 SSH 公钥。

11.1.7. 获取安装程序

在安装 OpenShift Container Platform 之前，将安装文件下载到本地计算机上。

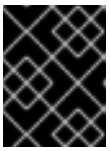
先决条件

- 运行 Linux 或 macOS 的计算机，本地磁盘空间为 500 MB

流程

1. 访问 OpenShift Cluster Manager 站点的 Infrastructure Provider 页面。如果您有红帽帐号，请使用自己的凭证登录。如果没有，请创建一个帐户。
2. 选择您的基础架构供应商。

3. 进入适用于您的安装类型的页面，下载您的操作系统的安装程序，并将文件放在要保存安装配置文件的目录中。。



重要

安装程序会在用来安装集群的计算机上创建若干文件。在完成集群安装后，您必须保留安装程序和安装程序所创建的文件。这两个文件都需要删除集群。

重要

删除安装程序创建的文件不会删除您的集群，即使集群在安装过程中失败也是如此。要删除集群，为特定云供应商完成 OpenShift Container Platform 卸载流程。

4. 提取安装程序。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ tar xvf openshift-install-linux.tar.gz
```

5. 从 Red Hat OpenShift Cluster Manager 下载安装 pull secret 。通过此 pull secret，您可以进行所含授权机构提供的服务的身份验证，这些服务包括为 OpenShift Container Platform 组件提供容器镜像的 Quay.io。

11.1.8. 在您的系统信任中添加 vCenter root CA 证书

由于安装程序需要访问 vCenter 的 API，所以必须在安装 OpenShift Container Platform 集群前将 vCenter 的可信 root CA 证书添加到系统信任中。

流程

1. 在 vCenter 主页中下载 vCenter 的 root CA 证书。在 vSphere Web Services SDK 部分点击 **Download trusted root CA certificates**。**<vCenter>/certs/download.zip** 文件下载。

2. 提取包含 vCenter root CA 证书的压缩文件。压缩文件的内容类似以下文件结构：

```
certs
├── lin
│   ├── 108f4d17.0
│   ├── 108f4d17.r1
│   ├── 7e757f6a.0
│   ├── 8e4f8471.0
│   └── 8e4f8471.r0
├── mac
│   ├── 108f4d17.0
│   ├── 108f4d17.r1
│   ├── 7e757f6a.0
│   ├── 8e4f8471.0
│   └── 8e4f8471.r0
└── win
    ├── 108f4d17.0.crt
    ├── 108f4d17.r1.crl
    ├── 7e757f6a.0.crt
    ├── 8e4f8471.0.crt
    └── 8e4f8471.r0.crl

3 directories, 15 files
```

3. 将您的操作系统的文件添加到系统信任中。例如，在 Fedora 操作系统上运行以下命令：

```
# cp certs/lin/* /etc/pki/ca-trust/source/anchors
```

4. 更新您的系统信任关系。例如，在 Fedora 操作系统上运行以下命令：

```
# update-ca-trust extract
```

11.1.9. 部署集群

您可以在兼容云平台中安装 OpenShift Container Platform。

重要

安装程序的 **create cluster** 命令只能在初始安装过程中运行一次。

先决条件

- 获取 OpenShift Container Platform 安装程序以及集群的 pull secret。

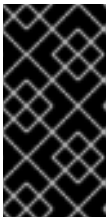
流程

1. 更改为包含安装程序的目录并初始化集群部署：

```
$ ./openshift-install create cluster --dir <installation_directory> \ 1
    --log-level=info 2
```

1 对于 **<installation_directory>**，请指定用于保存安装程序所创建的文件的目录名称。

2 要查看不同的安装详情，请指定 **warn**、**debug** 或 **error**，而不要指定 **info**。



重要

指定一个空目录。一些安装信息，如 bootstrap X.509 证书，有较短的过期间隔，因此不要重复使用安装目录。如果要重复使用另一个集群安装中的个别文件，可以将其复制到您的目录中。但是，一些安装数据的文件名可能会在发行版本之间有所改变。从 OpenShift Container Platform 老版本中复制安装文件时要格外小心。

在提示符处提供值：

a. 可选：选择用来访问集群机器的 SSH 密钥。

注意

对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 **ssh-agent** 进程使用的 SSH 密钥。

b. 选择 **vsphere** 作为目标平台。

c. 指定 vCenter 实例的名称。

d. 指定创建集群所需的权限的 vCenter 帐户的用户名和密码。
安装程序连接到您的 vCenter 实例。

e. 选择要连接的 vCenter 实例中的数据中心。

f. 选择要使用的默认 vCenter 数据存储。

注意

数据存储和集群名称不能超过 60 个字符，因此请确保组合字符串长度不超过 60 个字符的限制。

g. 选择要在其中安装 vCenter 集群的 OpenShift Container Platform 集群。安装程序使用 vSphere 集群的 root 资源池作为默认资源池。

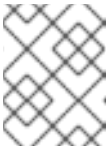
h. 选择包含您配置的虚拟 IP 地址和 DNS 记录的 vCenter 实例中的网络。

i. 输入您为 control plane API 访问配置的虚拟 IP 地址。

j. 输入您为集群入口配置的虚拟 IP 地址。

k. 输入基域。这个基域必须与您配置的 DNS 记录中使用的域相同。

l. 为集群输入一个描述性名称。集群名称必须与您配置的 DNS 记录中使用的相同。



注意

数据存储和集群名称不能超过 60 个字符，因此请确保组合字符串长度不超过 60 个字符的限制。

m. 粘贴 Red Hat OpenShift Cluster Manager 中的 pull secret 。

+ 集群部署完成后，终端会显示访问集群的信息，包括指向其 Web 控制台的链接和 **kubeadmin** 用户的凭证。

+ .输出示例

```
...
INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run 'export
KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'
INFO Access the OpenShift web-console here: https://console-openshift-
console.apps.mycluster.example.com
INFO Login to the console with user: "kubeadmin", and password: "4vYBz-Ee6gm-ymBZj-Wt5AL"
INFO Time elapsed: 36m22s
```

+

注意

当安装成功时，集群访问和凭证信息还会输出到 **<installation_directory>/.openshift_install.log**。

+

重要

- 安装程序生成的 Ignition 配置文件包含在 24 小时后过期的证书，然后在过期时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外情况是，您需要手动批准待处理的 **node-bootstrapper** 证书签名请求（CSR）来恢复 kubelet 证书。如需更多信息，请参阅*从过期的 control plane 证书中恢复*的文档。
- 建议您在生成 12 小时后使用 Ignition 配置文件，因为集群安装后 24 小时证书从 16 小时轮转至 22 小时。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中运行证书更新时避免安装失败。

+

重要

您不得删除安装程序或安装程序所创建的文件。需要这两者才能删除集群。

11.1.10. 通过下载二进制文件安装 OpenShift CLI

您需要安装 CLI（**oc**） 来使用命令行界面与 OpenShift Container Platform 进行交互。您可在 Linux 、Windows 或 macOS 上安装 **oc**。

重要

如果安装了旧版本的 **oc**，则无法使用 OpenShift Container Platform 4.6 中的所有命令。下载并安装新版本的 **oc**。

11.1.10.1. 在 Linux 上安装 OpenShift CLI

您可以按照以下流程在 Linux 上安装 OpenShift CLI（**oc**）二进制文件。

流程

1. 进入到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 在 **Version** 下拉菜单中选择相应的版本。
3. 单击 **OpenShift v4.6 Linux 客户端** 条目旁边的 **Download Now**，再保存文件。
4. 解包存档：

```
$ tar xvzf <file>
```

5. 把 **oc** 二进制代码放到 **PATH** 中的目录中。
   执行以下命令可以查看当前的 **PATH** 设置：

```
$ echo $PATH
```

安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
$ oc <command>
```

11.1.10.2. 在 Windows 上安装 OpenShift CLI

您可以按照以下流程在 Windows 上安装 OpenShift CLI（**oc**）二进制代码。

流程

1. 进入到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 在 **Version** 下拉菜单中选择相应的版本。
3. 单击 **OpenShift v4.6 Windows 客户端**条目旁边的 **Download Now**，再保存文件。
4. 使用 ZIP 程序解压存档。
5. 把 **oc** 二进制代码放到 **PATH** 中的目录中。
   要查看您的 **PATH**，请打开命令提示窗口并执行以下命令：

```
C:\> path
```

安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
C:\> oc <command>
```

11.1.10.3. 在 macOS 上安装 OpenShift CLI

您可以按照以下流程在 macOS 上安装 OpenShift CLI（**oc**）二进制代码。

流程

1. 进入到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 在 **Version** 下拉菜单中选择相应的版本。
3. 单击 **OpenShift v4.6 MacOSX 客户端**条目旁边的 **Download Now**，再保存文件。
4. 解包和解压存档。
5. 将 **oc** 二进制文件移到 PATH 的目录中。
   要查看您的 **PATH**，打开一个终端窗口并执行以下命令：

```
$ echo $PATH
```

安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
$ oc <command>
```

11.1.11. 使用 CLI 登录到集群

您可以通过导出集群 **kubeconfig** 文件，以默认系统用户身份登录集群。**kubeconfig** 文件包含关于集群的信息，供 CLI 用于将客户端连接到正确集群和 API 服务器。该文件特只适用于一个特定的集群，在 OpenShift Container Platform 安装过程中创建。

先决条件

- 已部署了 OpenShift Container Platform 集群。
- 已安装 **oc** CLI。

流程

1. 导出 **kubeadmin** 凭证：

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

2. 使用导出的配置，验证能否成功运行 **oc** 命令：

```
$ oc whoami
```

输出示例

```
system:admin
```

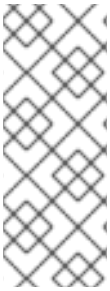
11.1.12. 创建 registry 存储

安装集群后，必须为 registry Operator 创建存储。

11.1.12.1. 安装过程中删除的镜像 registry

在不提供可共享对象存储的平台上，OpenShift Image Registry Operator bootstraps 本身的状态是 **Removed**。这允许 **openshift-installer** 在这些平台类型上完成安装。

将 **ManagementState** Image Registry Operator 配置从 **Removed** 改为 **Managed**。



注意

Prometheus 控制台提供了一个 **ImageRegistryRemoved** 警报，例如：

"Image Registry has been removed.**ImageStreamTags**, **BuildConfigs** and **DeploymentConfigs** which reference **ImageStreamTags** may not work as expected.Please configure storage and update the config to **Managed** state by editing configs.imageregistry.operator.openshift.io."

11.1.12.2. 镜像 registry 存储配置

对于不提供默认存储的平台，Image Registry Operator 最初将不可用。安装后，您必须配置 registry 使用的存储，这样 Registry Operator 才可用。

示配置生产集群所需的持久性卷的说明。如果适用，显示有关将空目录配置为存储位置的说明，该位置只可用于非生产集群。

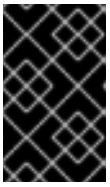
另外还提供了在升级过程中使用 **Recreate** rollout 策略来允许镜像 registry 使用块存储类型的说明。

11.1.12.2.1. 为 VMware vSphere 配置 registry 存储

作为集群管理员，在安装后需要配置 registry 来使用存储。

先决条件

- 具有 Cluster Administrator 权限
- VMware vSphere上有一个集群。
- 为集群置备的持久性存储，如 Red Hat OpenShift Container Storage。



重要

如果您只有一个副本，OpenShift Container Platform 支持对镜像 registry 存储的 **ReadWriteOnce** 访问。要部署支持高可用性的、带有两个或多个副本的镜像 registry，需要 **ReadWriteMany** 访问设置。

- 必须有" 100Gi"容量。

重要

测试显示，在 RHEL 中使用 NFS 服务器作为核心服务的存储后端可能会出现问题。这包括 OpenShift Container Registry 和 Quay，Prometheus 用于监控存储，以及 Elasticsearch 用于日志存储。因此，不推荐使用 RHEL NFS 作为 PV 后端用于核心服务。

市场上的其他 NFS 实现可能没有这些问题。如需了解更多与此问题相关的信息，请联络相关的 NFS 厂商。

流程

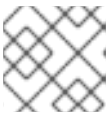
1. 为了配置 registry 使用存储，需要修改 **configs.imageregistry/cluster** 资源中的 **spec.storage.pvc** 。

注意

使用共享存储时，请查看您的安全设置以防止被外部访问。

2. 验证您没有 registry pod:

```
$ oc get pod -n openshift-image-registry
```



注意

如果存储类型为 **emptyDIR**，则副本数不能超过 **1**。

3. 检查 registry 配置：

```
$ oc edit configs.imageregistry.operator.openshift.io
```

输出示例

```
storage:
  pvc:
    claim: 1
```

1 将 **claim** 字段留空以允许自动创建一个 **image-registry-storage** PVC。

4. 检查 **clusteroperator** 的状态：

```
$ oc get clusteroperator image-registry
```

11.1.12.2.2. 为 VMware vSphere 配置块 registry 存储

在作为集群管理员升级时，要允许镜像 registry 使用块存储类型，如 vSphere Virtual Machine Disk（VMDK），您可以使用 **Recreate** rollout 策略。

重要

支持块存储卷，但不建议将其用于生产环境中的镜像 registry。在块存储上配置 registry 的安装不具有高可用性，因为 registry 无法拥有多个副本。

流程

1. 要将镜像 registry 存储设置为块存储类型，对 registry 进行补丁，使其使用 **Recreate** rollout 策略，且仅使用 **1** 个副本运行：

```
$ oc patch config.imageregistry.operator.openshift.io/cluster --type=merge -p '{"spec":
{"rolloutStrategy":"Recreate","replicas":1}}'
```

2. 为块存储设备置备 PV，并为该卷创建 PVC。请求的块卷使用 ReadWriteOnce（RWO）访问模式。

   a. 创建包含以下内容的 **pvc.yaml** 文件以定义 VMware vSphere **PersistentVolumeClaim**：

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: image-registry-storage 1
  namespace: openshift-image-registry 2
spec:
  accessModes:
  - ReadWriteOnce 3
  resources:
    requests:
      storage: 100Gi 4
```

1 代表 **PersistentVolumeClaim** 对象的唯一名称。

2 **PersistentVolumeClaim** 对象的命名空间，即 **openshift-image-registry**。

3 持久性卷声明的访问模式。使用 **ReadWriteOnce** 时，单个节点可以通过读写权限挂载这个卷。

4 持久性卷声明的大小。

b. 从文件创建 **PersistentVolumeClaim** 对象：

```
$ oc create -f pvc.yaml -n openshift-image-registry
```

3. 编辑 registry 配置，使其可以正确引用 PVC:

```
$ oc edit config.imageregistry.operator.openshift.io -o yaml
```

输出示例

```
storage:
  pvc:
    claim: 1
```

1 通过创建自定义 PVC，您可以将 **claim** 字段留空以用于默认自动创建 **image-registry-storage** PVC。

有关配置 registry 存储以便引用正确的 PVC 的说明，请参阅为 vSphere 配置 registry。

11.1.13. 备份 VMware vSphere 卷

OpenShift Container Platform 将新卷作为独立持久性磁盘置备，以便在集群中的任何节点上自由附加和分离卷。因此，无法备份使用快照的卷，也无法从快照中恢复卷。如需更多信息，请参阅 快照限制。

流程

要创建持久性卷的备份：

1. 停止使用持久性卷的应用程序。
2. 克隆持久性卷。
3. 重启应用程序。
4. 创建克隆的卷的备份。
5. 删除克隆的卷。

11.1.14. OpenShift Container Platform 的 Telemetry 访问

在 OpenShift Container Platform 4.6 中，默认运行的 Telemetry 服务提供有关集群健康状况和成功更新的指标，需要访问互联网。如果您的集群连接到互联网，Telemetry 会自动运行，而且集群会注册到 OpenShift Cluster Manager。

确认 OpenShift Cluster Manager 清单正确后，可以由 Telemetry 自动维护，也可以使用 OpenShift Cluster Manager 手动维护，使用订阅监控来跟踪帐户或多集群级别的 OpenShift Container Platform 订阅。

其他资源

- 有关 Telemetry 服务的更多信息，请参阅关于远程健康监控。

11.1.15. 后续步骤

- 自定义集群。
- 如果需要，您可以选择不使用远程健康报告。
- 设置 registry 并配置 registry 存储。

11.2. 在 VSPHERE 上安装自定义集群

在 OpenShift Container Platform 版本 4.6 中，您可以使用安装程序置备的基础架构在 VMware vSphere 实例上安装集群。要自定义安装，请在安装集群前修改 **install-config.yaml** 文件中的参数。

注意

OpenShift Container Platform 支持将集群部署到单个 VMware vCenter 中。不支持在多个 vCenter 上使用机器/机器集部署集群。

11.2.1. 先决条件

- 为集群置备持久性存储。若要部署私有镜像 registry，您的存储必须提供 **ReadWriteMany** 访问模式。
- 查看有关 OpenShift Container Platform 安装和更新流程的详细信息。
- OpenShift Container Platform 安装程序需要访问 vCenter 和 ESXi 主机上的端口 443。您确认可以访问端口 443。
- 如果您使用防火墙，您与管理员确认可以访问端口 443。control plane 节点必须能够通过端口 443 访问 vCenter 和 ESXi 主机，才能成功安装。
- 如果使用防火墙，则必须将其配置为允许集群需要访问的站点。

注意

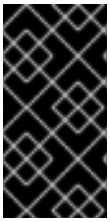
如果您要配置代理，请务必也要查看此站点列表。

11.2.2. OpenShift Container Platform 的互联网访问

在 OpenShift Container Platform 4.6 中，您需要访问互联网来安装集群。

您必须具有以下互联网访问权限：

- 访问 OpenShift Cluster Manager 以下载安装程序并执行订阅管理。如果集群可以访问互联网，并且没有禁用 Telemetry，该服务会自动授权您的集群。
- 访问 Quay.io，以获取安装集群所需的软件包。
- 获取执行集群更新所需的软件包。



重要

如果您的集群无法直接访问互联网，则可以在置备的某些类基础架构上执行受限网络安装。在此过程中，您要下载所需的内容，并使用它在镜像 registry（mirror registry） 中填充安装集群并生成安装程序所需的软件包。对于某些安装类型，集群要安装到的环境不需要访问互联网。在更新集群之前，要更新 registry 镜像系统中的内容。

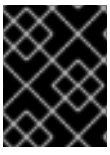
11.2.3. VMware vSphere 基础架构要求

您必须在满足您使用的组件要求的 VMware vSphere 版本 6 或 7 实例上安装 OpenShift Container Platform 集群。

表 11.6. VMware 组件支持的最低 vSphere 版本

组件	最低支持版本	描述
虚拟机监控程序	vSphere 6.5 及之后的版本 13	此版本是 Red Hat Enterprise Linux CoreOS(RHCOS)支持的最低版本。请查看 Red Hat Enterprise Linux 8 支持的管理程序列表。
使用 in-tree 驱动程序存储	vSphere 6.5 及之后的版本	此插件使用 OpenShift Container Platform 中包含的 vSphere 的树内存储驱动程序创建 vSphere 存储。
可选： Networking (NSX-T)	vSphere 6.5U3 或 vSphere 6.7U2 及之后的版本	OpenShift Container Platform 需要 vSphere 6.5U3 或 vSphere 6.7U2+。VMware 的 NSX Container Plug-in（NCP）3.0.2 使用 OpenShift Container Platform 4.6 和 NSX-T 3.x+ 认证。

如果您使用 vSphere 版本 6.5 实例，请在安装 OpenShift Container Platform 前考虑升级到 6.7U3 或 7.0。



重要

您必须确保在安装 OpenShift Container Platform 前同步 ESXi 主机上的时间。请参阅 VMware 文档中的编辑主机时间配置。

11.2.4. 网络连接要求

您必须配置机器之间的网络连接，以允许 OpenShift Container Platform 集群组件进行通信。

查看有关所需网络端口的以下详细信息。

表 11.7. 用于全机器到所有机器通信的端口

协议	端口	描述
ICMP	N/A	网络可访问性测试
TCP	**1936**	指标
	9000-9999	主机级别的服务，包括端口 **9100-9101** 上的节点导出器和端口 **9099** 上的 Cluster Version Operator。
	10250-10259	Kubernetes 保留的默认端口
	10256	openshift-sdn
UDP	**4789**	虚拟可扩展 LAN(VXLAN)
	6081	Geneve
	9000-9999	主机级别的服务，包括端口 **9100-9101** 上的节点导出器。
	500	IPsec IKE 数据包
	4500	IPsec NAT-T 数据包
TCP/UDP	**30000-32767**	Kubernetes 节点端口
ESP	N/A	IPsec Encapsulating Security Payload(ESP)

表 11.8. 用于所有机器控制平面通信的端口

协议	端口	描述
TCP	**6443**	Kubernetes API

表 11.9. control plane 机器用于 control plane 机器通信的端口

协议	端口	描述
TCP	**2379-2380**	etcd 服务器和对等端口

11.2.5. vCenter 要求

在使用安装程序置备的基础架构的 vCenter 上安装 OpenShift Container Platform 集群前，您必须准备自己的环境。

所需的 vCenter 帐户权限
要在 vCenter 中安装 OpenShift Container Platform 集群，安装程序需要一个具有特权的帐户来读取和创建所需资源。使用具有全局管理特权的帐户是访问所有必要权限的最简单方式。

如果无法使用具有全局管理特权的帐户，您必须创建角色来授予 OpenShift Container Platform 集群安装所需的权限。虽然大多数权限始终是必需的，但是一些权限只有在计划安装程序需要在您的 vCenter 实例中置备一个包含 OpenShift Container Platform 集群的文件夹时（这是默认行为）才需要。您必须为指定对象创建或修改 vSphere 角色，才能授予所需的权限。

如果安装程序创建 vSphere 虚拟机文件夹，则需要额外的角色。

例 11.3. 安装所需的角色和权限

角色的 vSphere 对象	何时需要	所需的权限
vSphere vCenter	Always	**Cns.Searchable** **InventoryService.Tagging.AttachTag** **InventoryService.Tagging.CreateCategory** **InventoryService.Tagging.CreateTag** **InventoryService.Tagging.DeleteCategory** **InventoryService.Tagging.DeleteTag** **InventoryService.Tagging.EditCategory** **InventoryService.Tagging.EditTag** **Sessions.ValidateSession** **StorageProfile.View**
vSphere vCenter Cluster	Always	**Host.Config.Storage** **Resource.AssignVMToPool** **VApp.AssignResourcePool** **VApp.Import** **VirtualMachine.Config.AddNewDisk**
vSphere Datastore	Always	**Datastore.AllocateSpace** **Datastore.Browse** **Datastore.FileManagement**
vSphere 端口组	Always	**Network.Assign**

角色的 vSphere 对象	何时需要	所需的权限
虚拟机文件夹	Always	**Resource.AssignVMToPool VApp.Import VirtualMachine.Config.Add ExistingDisk VirtualMachine.Config.Add NewDisk VirtualMachine.Config.Add RemoveDevice VirtualMachine.Config.Adva ncedConfig VirtualMachine.Config.Anno tation VirtualMachine.Config.CPU Count VirtualMachine.Config.Disk Extend VirtualMachine.Config.Disk Lease VirtualMachine.Config.Edit Device VirtualMachine.Config.Mem ory VirtualMachine.Config.Rem oveDisk VirtualMachine.Config.Rena me VirtualMachine.Config.Rese tGuestInfo VirtualMachine.Config.Reso urce VirtualMachine.Config.Setti ngs VirtualMachine.Config.Upgr adeVirtualHardware VirtualMachine.Interact.Gue stControl VirtualMachine.Interact.Pow erOff VirtualMachine.Interact.Pow erOn VirtualMachine.Interact.Res et VirtualMachine.Inventory.Cr eate VirtualMachine.Inventory.Cr eateFromExisting VirtualMachine.Inventory.D elete VirtualMachine.Provisionin g.Clone**

角色的 vSphere 对象	何时需要	所需的权限
vSphere vCenter Datacenter	如果安装程序创建虚拟机文件夹	**Resource.AssignVMToPool VApp.Import VirtualMachine.Config.Add ExistingDisk VirtualMachine.Config.Add NewDisk VirtualMachine.Config.Add RemoveDevice VirtualMachine.Config.Adva ncedConfig VirtualMachine.Config.Anno tation VirtualMachine.Config.CPU Count VirtualMachine.Config.Disk Extend VirtualMachine.Config.Disk Lease VirtualMachine.Config.Edit Device VirtualMachine.Config.Mem ory VirtualMachine.Config.Rem oveDisk VirtualMachine.Config.Rena me VirtualMachine.Config.Rese tGuestInfo VirtualMachine.Config.Reso urce VirtualMachine.Config.Setti ngs VirtualMachine.Config.Upgr adeVirtualHardware VirtualMachine.Interact.Gue stControl VirtualMachine.Interact.Pow erOff VirtualMachine.Interact.Pow erOn VirtualMachine.Interact.Res et VirtualMachine.Inventory.Cr eate VirtualMachine.Inventory.Cr eateFromExisting VirtualMachine.Inventory.D elete VirtualMachine.Provisionin g.Clone Folder.Create Folder.Delete**

此外，用户需要一些 **ReadOnly** 权限，某些角色需要权限来提升对子对象的权限。这些设置会根据您是否将集群安装到现有文件夹而有所不同。

例 11.4. 所需的权限和传播设置

vSphere 对象	文件夹类型	传播到子对象	所需的权限
vSphere vCenter	Always	False	列出所需的权限
vSphere vCenter Datacenter	现有文件夹	False	**ReadOnly** 权限
	安装程序创建文件夹	True	列出所需的权限
vSphere vCenter Cluster	Always	True	列出所需的权限
vSphere vCenter Datastore	Always	False	列出所需的权限
vSphere Switch	Always	False	**ReadOnly** 权限
vSphere 端口组	Always	False	列出所需的权限
vSphere vCenter Virtual Machine Folder	现有文件夹	True	列出所需的权限

有关只使用所需权限创建帐户的更多信息，请参阅 vSphere 文档中的 vSphere 权限和用户管理任务。

将 OpenShift Container Platform 与 vMotion 搭配使用

如果要在 vSphere 环境中使用 vMotion，请在安装 OpenShift Container Platform 集群前考虑以下内容。

- OpenShift Container Platform 通常支持仅用于计算的 vMotion。使用 Storage vMotion 可能会导致问题且不被支持。
  为了帮助确保计算和 control plane 节点的正常运行时间，建议您遵循 VMware 最佳实践进行 vMotion。还建议使用 VMware 反关联性规则来改进 OpenShift Container Platform 在维护或硬件问题期间的可用性。

  有关 vMotion 和 anti-affinity 规则的更多信息，请参阅 VMware vSphere 文档 了解 vMotion 网络要求和虚拟机反关联性规则。

- 如果您在 pod 中使用 vSphere 卷，请手动或通过 Storage vMotion 在数据存储间迁移虚拟机，从而导致 OpenShift Container Platform 持久性卷(PV)对象中的无效引用。这些引用可防止受影响的 pod 启动，并可能导致数据丢失。

- 同样，OpenShift Container Platform 不支持在数据存储间有选择地迁移 VMDK、使用数据存储集群进行虚拟机置备、动态或静态置备 PV，或使用作为数据存储集群一部分的数据存储进行 PV 的动态或静态置备。

集群资源

当部署使用安装程序置备的基础架构的 OpenShift Container Platform 集群时，安装程序必须能够在 vCenter 实例中创建多个资源。

标准 OpenShift Container Platform 安装会创建以下 vCenter 资源：

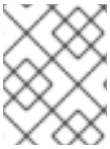
- 1 个文件夹
- 1 标签（Tag）类别
- 1 个标签（Tag）
- 虚拟机:
  - 1 个模板
  - 1 个临时 bootstrap 节点
  - 3 个 control plane 节点
  - 3 个计算机器

虽然这些资源使用了 856 GB 存储，但 bootstrap 节点会在集群安装过程中被销毁。使用标准集群至少需要 800 GB 存储。

如果部署了更多计算机器，OpenShift Container Platform 集群将使用更多存储。

集群的限制
可用资源因集群而异。vCenter 中可能的集群数量主要受可用存储空间以及对所需资源数量的限制。确保考虑集群创建的 vCenter 资源的限制和部署集群所需的资源，如 IP 地址和网络。

网络要求
网络必须使用 DHCP，并确保 DHCP 服务器被配置为为集群机器提供持久的 IP 地址。



注意

在安装开始前，持久性 IP 地址不可用。分配 DHCP 范围后，在安装后使用持久 IP 地址手动替换分配。

另外，在安装 OpenShift Container Platform 集群前，必须创建以下网络资源：

注意

建议集群中的每个 OpenShift Container Platform 节点都可以访问可通过 DHCP 发现的网络时间协议（NTP）服务器。没有 NTP 服务器也可安装。但是，异步服务器时钟将导致错误，NTP 服务器会阻止。

所需的 IP 地址
安装程序置备的 vSphere 安装需要这些静态 IP 地址：

- API 地址用于访问集群 API。
- Ingress 地址用于集群入口流量。
- 当将集群从版本 4.5 升级到 4.6 时，会使用 control plane 节点地址。

安装 OpenShift Container Platform 集群时，必须向安装程序提供这些 IP 地址。

DNS 记录

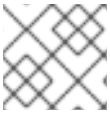
您必须在正确的 DNS 服务器中为托管 OpenShift Container Platform 集群的 vCenter 实例创建两个静态 IP 地址的 DNS 记录。在每个记录中，**<cluster_name>** 是集群名称，**<base_domain>** 是您在安装集群时指定的集群基域。完整的 DNS 记录采用如下格式: **<component>.<cluster_name>.<base_domain>.**。

表 11.10. 所需的 DNS 记录

组件	记录	描述
API VIP	**api.<cluster_name>.<base_domain>.**	此 DNS A/AAAA 或 CNAME 记录必须指向 control plane 机器的负载均衡器。此记录必须能由集群外的客户端和集群内的所有节点解析。
Ingress VIP	***.apps.<cluster_name>.<base_domain>.**	通配符 DNS A/AAAA 或 CNAME 记录，指向以运行入口路由器 Pod 的机器（默认为 worker 节点）为目标的负载均衡器。此记录必须能由集群外的客户端和集群内的所有节点解析。

11.2.6. 生成 SSH 私钥并将其添加到代理中

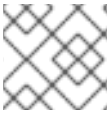
如果要在集群上执行安装调试或灾难恢复，则必须为 **ssh-agent** 和安装程序提供 SSH 密钥。您可以使用此密钥访问公共集群中的 bootstrap 机器来排除安装问题。



注意

在生产环境中，您需要进行灾难恢复和调试。

您可以使用此密钥以 **core** 用户身份通过 SSH 连接到 master 节点。在部署集群时，此密钥会添加到 **core** 用户的 **~/.ssh/authorized_keys** 列表中。



注意

您必须使用一个本地密钥，而不要使用在特定平台上配置的密钥，如 AWS 密钥对。

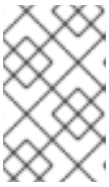
流程

1. 如果还没有为计算机上免密码身份验证而配置的 SSH 密钥，请创建一个。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ ssh-keygen -t ed25519 -N '' \
  -f <path>/<file_name> 1
```

1 指定新 SSH 密钥的路径和文件名，如 **~/.ssh/id_rsa**。如果您已有密钥对，请确保您的公钥位于 **~/.ssh** 目录中。

运行此命令会在指定的位置生成不需要密码的 SSH 密钥。



注意

如果您计划在 **x86_64** 架构中安装使用 FIPS 验证的/Modules in Process 加密库的 OpenShift Container Platform 集群，不要创建使用 **ed25519** 算法的密钥。反之，创建一个使用 **rsa** 或 **ecdsa** 算法的密钥。

2. 作为后台任务启动 **ssh-agent** 进程：

```
$ eval "$(ssh-agent -s)"
```

输出示例

```
Agent pid 31874
```

注意

如果您的集群采用 FIPS 模式，则只使用 FIPS 兼容算法来生成 SSH 密钥。密钥必须是 RSA 或 ECDSA。

3. 将 SSH 私钥添加到 **ssh-agent**：

```
$ ssh-add <path>/<file_name> 1
```

输出示例

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

1 指定 SSH 私钥的路径和文件名，如 **~/.ssh/id_rsa**

后续步骤

- 在安装 OpenShift Container Platform 时，为安装程序提供 SSH 公钥。

11.2.7. 获取安装程序

在安装 OpenShift Container Platform 之前，将安装文件下载到本地计算机上。

先决条件

- 运行 Linux 或 macOS 的计算机，本地磁盘空间为 500 MB

流程

1. 访问 OpenShift Cluster Manager 站点的 Infrastructure Provider 页面。如果您有红帽帐号，请使用自己的凭证登录。如果没有，请创建一个帐户。
2. 选择您的基础架构供应商。
3. 进入适用于您的安装类型的页面，下载您的操作系统的安装程序，并将文件放在要保存安装配置文件的目录中。。

重要

安装程序会在用来安装集群的计算机上创建若干文件。在完成集群安装后，您必须保留安装程序和安装程序所创建的文件。这两个文件都需要删除集群。

重要

删除安装程序创建的文件不会删除您的集群，即使集群在安装过程中失败也是如此。要删除集群，为特定云供应商完成 OpenShift Container Platform 卸载流程。

4. 提取安装程序。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ tar xvf openshift-install-linux.tar.gz
```

5. 从 Red Hat OpenShift Cluster Manager 下载安装 pull secret 。通过此 pull secret，您可以进行所含授权机构提供的服务的身份验证，这些服务包括为 OpenShift Container Platform 组件提供容器镜像的 Quay.io。

11.2.8. 在您的系统信任中添加 vCenter root CA 证书

由于安装程序需要访问 vCenter 的 API，所以必须在安装 OpenShift Container Platform 集群前将 vCenter 的可信 root CA 证书添加到系统信任中。

流程

1. 在 vCenter 主页中下载 vCenter 的 root CA 证书。在 vSphere Web Services SDK 部分点击 **Download trusted root CA certificates**。**<vCenter>/certs/download.zip** 文件下载。

2. 提取包含 vCenter root CA 证书的压缩文件。压缩文件的内容类似以下文件结构：

```
certs
├── lin
│   ├── 108f4d17.0
│   ├── 108f4d17.r1
│   ├── 7e757f6a.0
│   ├── 8e4f8471.0
│   └── 8e4f8471.r0
├── mac
│   ├── 108f4d17.0
│   ├── 108f4d17.r1
│   ├── 7e757f6a.0
│   ├── 8e4f8471.0
│   └── 8e4f8471.r0
└── win
    ├── 108f4d17.0.crt
    ├── 108f4d17.r1.crl
    ├── 7e757f6a.0.crt
    ├── 8e4f8471.0.crt
    └── 8e4f8471.r0.crl

3 directories, 15 files
```

3. 将您的操作系统的文件添加到系统信任中。例如，在 Fedora 操作系统上运行以下命令：

```
# cp certs/lin/* /etc/pki/ca-trust/source/anchors
```

4. 更新您的系统信任关系。例如，在 Fedora 操作系统上运行以下命令：

```
# update-ca-trust extract
```

11.2.9. 创建安装配置文件

您可以自定义在 VMware vSphere 上安装的 OpenShift Container Platform 集群。

先决条件

- 获取 OpenShift Container Platform 安装程序以及集群的 pull secret。

流程

1. 创建 **install-config.yaml** 文件。
   a. 更改到包含安装程序的目录，再运行以下命令：

```
$ ./openshift-install create install-config --dir <installation_directory> 1
```

   1 对于 **<installation_directory>**，请指定用于保存安装程序所创建的文件的目录名称。

重要

指定一个空目录。一些安装信息，如 bootstrap X.509 证书，有较短的过期间隔，因此不要重复使用安装目录。如果要重复使用另一个集群安装中的个别文件，可以将其复制到您的目录中。但是，一些安装数据的文件名可能会在发行版本之间有所改变。从 OpenShift Container Platform 老版本中复制安装文件时要格外小心。

   b. 在提示符处，提供您的云的配置详情：
      i. 可选：选择用来访问集群机器的 SSH 密钥。

注意

对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 **ssh-agent** 进程使用的 SSH 密钥。

      ii. 选择 **vsphere** 作为目标平台。
      iii. 指定 vCenter 实例的名称。
      iv. 指定创建集群所需的权限的 vCenter 帐户的用户名和密码。
      安装程序连接到您的 vCenter 实例。
      v. 选择要连接的 vCenter 实例中的数据中心。
      vi. 选择要使用的默认 vCenter 数据存储。

vii. 选择要在其中安装 vCenter 集群的 OpenShift Container Platform 集群。安装程序使用 vSphere 集群的 root 资源池作为默认资源池。

viii. 选择包含您配置的虚拟 IP 地址和 DNS 记录的 vCenter 实例中的网络。

ix. 输入您为 control plane API 访问配置的虚拟 IP 地址。

x. 输入您为集群入口配置的虚拟 IP 地址。

xi. 输入基域。这个基域必须与您配置的 DNS 记录中使用的域相同。

xii. 为集群输入一个描述性名称。集群名称必须与您配置的 DNS 记录中使用的相同。

xiii. 粘贴 Red Hat OpenShift Cluster Manager 中的 pull secret 。

2. 修改 **install-config.yaml** 文件。您可以在**安装配置参数**部分中找到有关可用参数的更多信息。

3. 备份 **install-config.yaml** 文件，以便用于安装多个集群。

重要

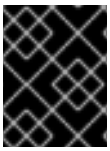
install-config.yaml 文件会在安装过程中消耗掉。如果要重复使用此文件，必须现在备份。

11.2.9.1. 安装配置参数

在部署 OpenShift Container Platform 集群前，您可以提供参数值，以描述托管集群的云平台的帐户并选择性地自定义集群平台。在创建 **install-config.yaml** 安装配置文件时，您可以通过命令行来提供所需的参数的值。如果要自定义集群，可以修改 **install-config.yaml** 文件来提供关于平台的更多信息。

注意

安装之后，您无法修改 **install-config.yaml** 文件中的这些参数。



重要

openshift-install 命令不验证参数的字段名称。如果指定了不正确的名称，则不会创建相关的文件或对象，且不会报告错误。确保所有指定的参数的字段名称都正确。

11.2.9.1.1. 所需的配置参数

下表描述了所需的安装配置参数：

表 11.11. 所需的参数

参数	描述	值
apiVersion	**install-config.yaml** 内容的 API 版本。当前版本是 **v1**。安装程序还可能支持旧的 API 版本。	字符串

参数	描述	值
baseDomain	云供应商的基域。此基础域用于创建到 OpenShift Container Platform 集群组件的路由。集群的完整 DNS 名称是 **baseDomain** 和 **metadata.name** 参数值的组合，其格式为 **<metadata.name>.<baseDomain>**。	完全限定域名或子域名，如 **example.com**。
metadata	Kubernetes 资源 **ObjectMeta**，其中只消耗 **name** 参数。	对象
metadata.name	集群的名称。集群的 DNS 记录是 **{{.metadata.name}}.{{.baseDomain}}** 的子域。	小写字母和连字符(**-**)的字符串，如 **dev**。
platform	执行安装的具体平台配置：**aws**、**baremetal**、**azure**、**openstack**、**ovirt**、**vsphere**。有关 **platform.<platform>** 参数的额外信息，请参考下表来了解您的具体平台。	对象

pullSecret	从 Red Hat OpenShift Cluster Manager 获取 pull secret，验证从 Quay.io 等服务中下载 OpenShift Container Platform 组件的容器镜像。	<pre>{
		  "auths":{
		  "cloud.openshift.com":{
		  "auth":"b3Blb=",
		  "email":"you@example.com"
		  },
		  "quay.io":{
		  "auth":"b3Blb=",
		  "email":"you@example.com"
		  }
		  }
		}</pre>

11.2.9.1.2. 网络配置参数

您可以根据现有网络基础架构的要求自定义安装配置。例如，您可以扩展集群网络的 IP 地址块，或者提供不同于默认值的不同 IP 地址块。

只支持 IPv4 地址。

表 11.12. 网络参数

参数	描述	值
networking	集群网络的配置。	对象 **注意** 您不能在安装后修改 **networking** 对象指定的参数。
networking.networkType	要安装的集群网络供应商 Container Network Interface（CNI）插件。	**OpenShiftSDN** 或 **OVNKubernetes**。默认值为 **OpenShiftSDN**。
networking.clusterNetwork	pod 的 IP 地址块。 默认值为 **10.128.0.0/14**，主机前缀为 **/23**。 如果您指定多个 IP 地址块，则块不得互相重叠。	一个对象数组。例如： networking: clusterNetwork: - cidr: 10.128.0.0/14 hostPrefix: 23
networking.clusterNetwork.cidr	使用 **networking.clusterNetwork** 时需要此项。IP 地址块。 一个 IPv4 网络。	使用 CIDR 形式的 IP 地址块。IPv4 块的前缀长度介于 **0** 到 **32** 之间。
networking.clusterNetwork.hostPrefix	分配给每个单独节点的子网前缀长度。例如，如果 **hostPrefix** 设为 **23**，则每个节点从所给的 **cidr** 中分配一个 **/23** 子网。**hostPrefix** 值 **23** 提供 510（2^(32 - 23)- 2）个 pod IP 地址。	子网前缀。 默认值为 **23**。
networking.serviceNetwork	服务的 IP 地址块。默认值为 **172.30.0.0/16**。 OpenShift SDN 和 OVN-Kubernetes 网络供应商只支持服务网络的一个 IP 地址块。	CIDR 格式具有 IP 地址块的数组。例如： networking: serviceNetwork: - 172.30.0.0/16
networking.machineNetwork	机器的 IP 地址块。 如果您指定多个 IP 地址块，则块不得互相重叠。	一个对象数组。例如： networking: machineNetwork: - cidr: 10.0.0.0/16

参数	描述	值
networking.machineNetwork.cidr	使用 **networking.machineNetwork** 时需要。IP 地址块。libvirt 以外的所有平台的默认值为 **10.0.0.0/16**。对于 libvirt，默认值为 **192.168.126.0/24**。	CIDR 表示法中的 IP 网络块。 例如： **10.0.0.0/16**。 **注意** 将 **networking.machineNetwork** 设置为与首选 NIC 所在的 CIDR 匹配。

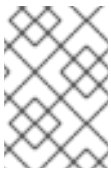
11.2.9.1.3. 可选配置参数

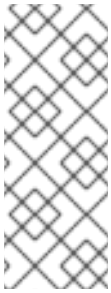
下表描述了可选安装配置参数：

表 11.13. 可选参数

参数	描述	值
additionalTrustBundle	添加到节点可信证书存储中的 PEM 编码 X.509 证书捆绑包。配置了代理时，也可以使用这个信任捆绑包。	字符串
compute	组成计算节点的机器的配置。	machine-pool 对象的数组。详情请查看以下"Machine-pool"表。
compute.architecture	决定池中机器的指令集合架构。目前不支持异构集群，因此所有池都必须指定相同的架构。有效值为 **amd64** （默认值）。	字符串
compute.hyperthreading	是否在计算机器上启用或禁用并发多线程或**超线程**。默认情况下，启用并发多线程以提高机器内核的性能。 **重要** 如果禁用并发多线程，请确保在容量规划时考虑到机器性能可能会显著降低的问题。	**Enabled** 或 **Disabled**
compute.name	使用 **compute** 时需要此值。机器池的名称。	**worker**

参数	描述	值
compute.platform	使用 **compute** 时需要此值。使用此参数指定托管 worker 机器的云供应商。此参数值必须与 **controlPlane.platform** 参数值匹配。	**aws**、**azure**、**gcp**、**openstack**、**ovirt**、**vsphere** 或 **{}**
compute.replicas	要置备的计算机器数量，也称为 worker 机器。	大于或等于 **2** 的正整数。默认值为 **3**。
controlPlane	组成 control plane 的机器的配置。	**MachinePool** 对象的数组。详情请查看以下"Machine-pool"表。
controlPlane.architecture	决定池中机器的指令集合架构。目前不支持异构集群，因此所有池都必须指定相同的架构。有效值为 **amd64** （默认值）。	字符串
controlPlane.hyperthreading	是否在 control plane 机器上启用或禁用并发多线程或**超线程**。默认情况下，启用并发多线程以提高机器内核的性能。  **重要** 如果禁用并发多线程，请确保在容量规划时考虑到机器性能可能会显著降低的问题。	**Enabled** 或 **Disabled**
controlPlane.name	使用 **controlPlane** 时需要。机器池的名称。	**master**
controlPlane.platform	使用 **controlPlane** 时需要。使用此参数指定托管 control plane 机器的云供应商。此参数值必须与 **compute.platform** 参数值匹配。	**aws**、**azure**、**gcp**、**openstack**、**ovirt**、**vsphere** 或 **{}**
controlPlane.replicas	要置备的 control plane 机器数量。	唯一支持的值是 **3**，它是默认值。

参数	描述	值
credentialsMode	Cloud Credential Operator（CCO）模式。如果没有指定任何模式，CCO 会动态地尝试决定提供的凭证的功能，在支持多个模式的平台上使用 mint 模式。  **注意** 不是所有 CCO 模式都支持所有云供应商。如需有关 CCO 模式的更多信息，请参阅*Red Hat Operator 参考指南*内容中的 *Cloud Credential Operator* 条目。	**Mint**、**Passthrough**、**Manual** 或空字符串(**""**)。
fips	启用或禁用 FIPS 模式。默认为 **false**（禁用）。如果启用了 FIPS 模式，运行 OpenShift Container Platform 的 Red Hat Enterprise Linux CoreOS (RHCOS) 机器会绕过默认的 Kubernetes 加密套件，并使用由 RHCOS 提供的加密模块。  **重要** 只有在 **x86_64** 架构中的 OpenShift Container Platform 部署支持 FIPS 验证的/Modules in Process 加密库。  **注意** 如果使用 Azure File 存储，则无法启用 FIPS 模式。	**false** 或 **true**
imageContentSources	release-image 内容的源和仓库。	对象数组。包括一个 **source** 以及可选的 **mirrors**，如下表所示。
imageContentSources.source	使用 **imageContentSources** 时需要。指定用户在镜像拉取规格中引用的仓库。	字符串
imageContentSources.mirrors	指定可能还包含同一镜像的一个或多个仓库。	字符串数组

参数	描述	值
publish	如何发布或公开集群的面向用户的端点，如 Kubernetes API、OpenShift 路由。	**Internal** 或 **External**。默认值为 **External**。 在非云平台上不支持将此字段设置为 **Internal**。  **重要** 如果将字段的值设为 **Internal**，集群将无法运行。如需更多信息，请参阅 BZ#1953035。
sshKey	用于验证集群机器访问的 SSH 密钥或密钥。  **注意** 对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 **ssh-agent** 进程使用的 SSH 密钥。	一个或多个密钥。例如： sshKey: <key1> <key2> <key3>

11.2.9.1.4. 其他 VMware vSphere 配置参数

下表描述了其他 VMware vSphere 配置参数：

表 11.14. 其他 VMware vSphere 集群参数

参数	描述	值
platform.vsphere.vCenter	vCenter 服务器的完全限定主机名或 IP 地址。	字符串
platform.vsphere.username	用于连接 vCenter 实例的用户名。此用户必须至少具有 vSphere 中 静态或动态持久性卷置备 所需的角色和权限。	字符串
platform.vsphere.password	vCenter 用户名的密码。	字符串

参数	描述	值
platform.vsphere.datacenter	要在 vCenter 实例中使用的数据中心的名称。	字符串
platform.vsphere.defaultDatastore	用于置备卷的默认数据存储名称。	字符串
platform.vsphere.folder	*可选*。安装程序创建虚拟机的现有文件夹的绝对路径。如果没有提供这个值，安装程序会创建一个文件夹，它的名称是数据中心虚拟机文件夹中的基础架构 ID。	字符串，如 **/<datacenter_name>/vm/<folder_name>/<subfolder_name>**。
platform.vsphere.network	包含您配置的虚拟 IP 地址和 DNS 记录的 vCenter 实例中的网络。	字符串
platform.vsphere.cluster	在其中安装 OpenShift Container Platform 集群的 vCenter 集群。	字符串
platform.vsphere.apiVIP	为 control plane API 访问配置的虚拟 IP（VIP）地址。	IP 地址，如 **128.0.0.1**。
platform.vsphere.ingressVIP	为集群入口配置的虚拟 IP（VIP）地址。	IP 地址，如 **128.0.0.1**。

11.2.9.1.5. 可选的 VMware vSphere 机器池配置参数

下表描述了可选的 VMware vSphere 机器池配置参数：

表 11.15. 可选的 VMware vSphere 机器池参数

参数	描述	值
platform.vsphere.clusterOSImage	安装程序从中下载 RHCOS 镜像的位置。您必须设置此参数以便在受限网络中执行安装。	HTTP 或 HTTPS URL，可选使用 SHA-256 checksum。例如：**https://mirror.openshift.com/images/rhcos-<version>-vmware.<architecture>.ova**。
platform.vsphere.osDisk.diskSizeGB	以 GB 为单位的磁盘大小。	整数
platform.vsphere.cpus	分配虚拟机的虚拟处理器内核总数。	整数

参数	描述	值
platform.vsphere.coresPerSocket	虚拟机中每个插槽的内核数。虚拟机上的虚拟套接字数量为 **platform.vsphere.cpus/platform.vsphere.coresPerSocket**。默认值为 **1**。	整数
platform.vsphere.memoryMB	以 MB 为单位的虚拟机内存大小。	整数

11.2.9.2. 安装程序置备的 VMware vSphere 集群的 install-config.yaml 文件示例

您可以自定义 install-config.yaml 文件，以指定有关 OpenShift Container Platform 集群平台的更多信息，或修改所需参数的值。

```
apiVersion: v1
baseDomain: example.com (1)
compute: (2)
- hyperthreading: Enabled (3)
  name: worker
  replicas: 3
  platform:
    vsphere: (4)
      cpus: 2
      coresPerSocket: 2
      memoryMB: 8192
      osDisk:
        diskSizeGB: 120
controlPlane: (5)
  hyperthreading: Enabled (6)
  name: master
  replicas: 3
  platform:
    vsphere: (7)
      cpus: 4
      coresPerSocket: 2
      memoryMB: 16384
      osDisk:
        diskSizeGB: 120
metadata:
  name: cluster (8)
platform:
  vsphere:
    vcenter: your.vcenter.server
    username: username
    password: password
    datacenter: datacenter
    defaultDatastore: datastore
    folder: folder
    network: VM_Network
    cluster: vsphere_cluster_name (9)
```

```
    apiVIP: api_vip
    ingressVIP: ingress_vip
fips: false
pullSecret: '{"auths": ...}'
sshKey: 'ssh-ed25519 AAAA...'
```

1. 集群的基域。所有 DNS 记录都必须是这个基域的子域，并包含集群名称。

2 5. **controlPlane** 部分是一个单映射，但 **compute** 部分是一系列映射。为满足不同数据结构的要求，**compute** 部分的第一行必须以连字符 **-** 开头，**controlPlane** 部分的第一行则不可以连字符开头。只使用一个 control plane 池。

3 6. 是否要启用或禁用并发多线程或**超线程**。默认情况下，启用并发多线程以提高机器内核的性能。您可以通过将参数值设为 **Disabled** 来禁用。如果您在某些集群机器上禁用并发多线程，则必须在所有集群机器上禁用。

重要

如果禁用并发多线程，请确保在容量规划时考虑到机器性能可能会显著降低的问题。如果您禁用并发多线程，则计算机必须至少使用 8 个 CPU 和 32GB RAM。

4 7. 可选：为 compute 和 control plane 机器提供额外的机器池参数配置。

8. 您在 DNS 记录中指定的集群名称。

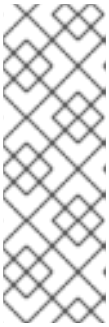
9. 要在其中安装 OpenShift Container Platform 集群的 vSphere 集群。安装程序使用 vSphere 集群的 root 资源池作为默认资源池。

11.2.9.3. 在安装过程中配置集群范围代理

生产环境可能会拒绝直接访问互联网，而是提供 HTTP 或 HTTPS 代理。您可以通过在 **install-config.yaml** 文件中配置代理设置，将新的 OpenShift Container Platform 集群配置为使用代理。

先决条件

- 您有一个现有的 **install-config.yaml** 文件。
- 您检查了集群需要访问的站点，并决定是否需要绕过代理。默认情况下代理所有集群出口流量，包括对托管云供应商 API 的调用。您需要将站点添加到 **Proxy** 对象的 **spec.noProxy** 字段来绕过代理。



注意

Proxy 对象 **status.noProxy** 字段使用安装配置中的 **networking.machineNetwork[].cidr**、**networking.clusterNetwork[].cidr** 和 **networking.serviceNetwork[]** 字段的值填充。

对于在 Amazon Web Services(AWS)、Google Cloud Platform(GCP)、Microsoft Azure 和 Red Hat OpenStack Platform(RHOSP)上安装，**Proxy** 对象 **status.noProxy** 字段也会使用实例元数据端点填充(**169.254.169.254**)。

流程

1. 编辑 **install-config.yaml** 文件并添加代理设置。例如：

```
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: example.com 3
additionalTrustBundle: | 4
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
...
```

1. 用于创建集群外 HTTP 连接的代理 URL。URL 必须是 **http**。
2. 用于创建集群外 HTTPS 连接的代理 URL。
3. 要排除在代理中的目标域名、IP 地址或其他网络 CIDR 的逗号分隔列表。在域前面加 **.** 来仅匹配子域。例如： **.y.com** 匹配 **x.y.com**，但不匹配 **y.com**。使用 ***** 绕过所有目的地的代理。您必须包含 vCenter 的 IP 地址以及用于其机器的 IP 范围。
4. 如果提供，安装程序会在 **openshift-config 命名空间中生成名为 user-ca- bundle** 的配置映射来保存额外的 CA 证书。如果您提供 **additionalTrustBundle** 和至少一个代理设置，则 **Proxy** 对象会被配置为引用 **trustedCA** 字段中的 **user-ca-bundle** 配置映射。然后，Cluster Network Operator 会创建一个 **trusted-ca-bundle** 配置映射，该配置映射将为 **trustedCA** 参数指定的内容与 RHCOS 信任捆绑包合并。 **additionalTrustBundle** 字段是必需的，除非代理的身份证书由来自 RHCOS 信任捆绑包的颁发机构签名。

注意

安装程序不支持代理的 **readinessEndpoints** 字段。

2. 保存该文件，并在安装 OpenShift Container Platform 时引用。

安装程序会创建一个名为 **cluster** 的集群范围代理，该代理使用提供的 **install-config.yaml** 文件中的代理设置。如果没有提供代理设置，仍然会创建一个 **cluster Proxy** 对象，但它会有一个空 **spec**。

注意

只支持名为 **cluster** 的 **Proxy** 对象，且无法创建额外的代理。

11.2.10. 部署集群

您可以在兼容云平台中安装 OpenShift Container Platform。

重要

安装程序的 **create cluster** 命令只能在初始安装过程中运行一次。

先决条件

- 获取 OpenShift Container Platform 安装程序以及集群的 pull secret。

流程

1. 更改为包含安装程序的目录并初始化集群部署：

```
$ ./openshift-install create cluster --dir <installation_directory> \ 1
    --log-level=info 2
```

1 对于 **<installation_directory>**，请指定自定义 **./install-config.yaml** 文件的位置。

2 要查看不同的安装详情，请指定 **warn**、**debug** 或 **error**，而不要指定 **info**。

集群部署完成后，终端会显示访问集群的信息，包括指向其 Web 控制台的链接和 **kubeadmin** 用户的凭证。

```
...
INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run 'export
KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'
INFO Access the OpenShift web-console here: https://console-openshift-
console.apps.mycluster.example.com
INFO Login to the console with user: "kubeadmin", and password: "4vYBz-Ee6gm-ymBZj-Wt5AL"
INFO Time elapsed: 36m22s
```

+

注意

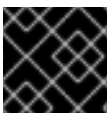
当安装成功时，集群访问和凭证信息还会输出到 **<installation_directory>/.openshift_install.log**。

+

重要

- 安装程序生成的 Ignition 配置文件包含在 24 小时后过期的证书，然后在过期时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外情况是，您需要手动批准待处理的 **node-bootstrapper** 证书签名请求（CSR）来恢复 kubelet 证书。如需更多信息，请参阅 *从过期的 control plane 证书中恢复*的文档。
- 建议您在生成 12 小时后使用 Ignition 配置文件，因为集群安装后 24 小时证书从 16 小时轮转至 22 小时。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中运行证书更新时避免安装失败。

+

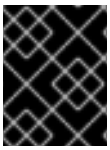


重要

您不得删除安装程序或安装程序所创建的文件。需要这两者才能删除集群。

11.2.11. 通过下载二进制文件安装 OpenShift CLI

您需要安装 CLI（**oc**） 来使用命令行界面与 OpenShift Container Platform 进行交互。您可在 Linux 、Windows 或 macOS 上安装 **oc**。



重要

如果安装了旧版本的 **oc**，则无法使用 OpenShift Container Platform 4.6 中的所有命令。下载并安装新版本的 **oc**。

11.2.11.1. 在 Linux 上安装 OpenShift CLI

您可以按照以下流程在 Linux 上安装 OpenShift CLI（**oc**）二进制文件。

流程

1. 进入到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 在 **Version** 下拉菜单中选择相应的版本。
3. 单击 **OpenShift v4.6 Linux 客户端** 条目旁边的 **Download Now**，再保存文件。
4. 解包存档：

```
$ tar xvzf <file>
```

5. 把 **oc** 二进制代码放到 **PATH** 中的目录中。
执行以下命令可以查看当前的 **PATH** 设置：

```
$ echo $PATH
```

安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
$ oc <command>
```

11.2.11.2. 在 Windows 上安装 OpenShift CLI

您可以按照以下流程在 Windows 上安装 OpenShift CLI（**oc**）二进制代码。

流程

1. 进入到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 在 **Version** 下拉菜单中选择相应的版本。
3. 单击 **OpenShift v4.6 Windows 客户端**条目旁边的 **Download Now**，再保存文件。
4. 使用 ZIP 程序解压存档。
5. 把 **oc** 二进制代码放到 **PATH** 中的目录中。
要查看您的 **PATH**，请打开命令提示窗口并执行以下命令：

```
C:\> path
```

安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
C:\> oc <command>
```

11.2.11.3. 在 macOS 上安装 OpenShift CLI

您可以按照以下流程在 macOS 上安装 OpenShift CLI（**oc**）二进制代码。

流程

1. 进入到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 在 **Version** 下拉菜单中选择相应的版本。
3. 单击 **OpenShift v4.6 MacOSX 客户端**条目旁边的 **Download Now**，再保存文件。
4. 解包和解压存档。
5. 将 **oc** 二进制文件移到 PATH 的目录中。
   要查看您的 **PATH**，打开一个终端窗口并执行以下命令：

   ```
   $ echo $PATH
   ```

安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
$ oc <command>
```

11.2.12. 使用 CLI 登录到集群

您可以通过导出集群 **kubeconfig** 文件，以默认系统用户身份登录集群。**kubeconfig** 文件包含关于集群的信息，供 CLI 用于将客户端连接到正确集群和 API 服务器。该文件特只适用于一个特定的集群，在 OpenShift Container Platform 安装过程中创建。

先决条件

- 已部署了 OpenShift Container Platform 集群。
- 已安装 **oc** CLI。

流程

1. 导出 **kubeadmin** 凭证：

   ```
   $ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
   ```

   1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

2. 使用导出的配置，验证能否成功运行 **oc** 命令：

   ```
   $ oc whoami
   ```

   输出示例

```
system:admin
```

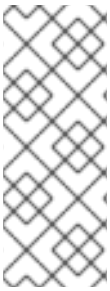
11.2.13. 创建 registry 存储

安装集群后，必须为 registry Operator 创建存储。

11.2.13.1. 安装过程中删除的镜像 registry

在不提供可共享对象存储的平台上，OpenShift Image Registry Operator bootstraps 本身的状态是 **Removed**。这允许 **openshift-installer** 在这些平台类型上完成安装。

将 **ManagementState** Image Registry Operator 配置从 **Removed** 改为 **Managed**。



注意

Prometheus 控制台提供了一个 **ImageRegistryRemoved** 警报，例如：

"Image Registry has been removed.**ImageStreamTags**, **BuildConfigs** and **DeploymentConfigs** which reference **ImageStreamTags** may not work as expected.Please configure storage and update the config to **Managed** state by editing configs.imageregistry.operator.openshift.io."

11.2.13.2. 镜像 registry 存储配置

对于不提供默认存储的平台，Image Registry Operator 最初将不可用。安装后，您必须配置 registry 使用的存储，这样 Registry Operator 才可用。

示配置生产集群所需的持久性卷的说明。如果适用，显示有关将空目录配置为存储位置的说明，该位置只可用于非生产集群。

另外还提供了在升级过程中使用 **Recreate** rollout 策略来允许镜像 registry 使用块存储类型的说明。

11.2.13.2.1. 为 VMware vSphere 配置 registry 存储

作为集群管理员，在安装后需要配置 registry 来使用存储。

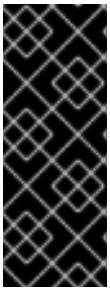
先决条件

- 具有 Cluster Administrator 权限
- VMware vSphere上有一个集群。
- 为集群置备的持久性存储，如 Red Hat OpenShift Container Storage。

重要

如果您只有一个副本，OpenShift Container Platform 支持对镜像 registry 存储的 **ReadWriteOnce** 访问。要部署支持高可用性的、带有两个或多个副本的镜像 registry，需要 **ReadWriteMany** 访问设置。

- 必须有“ 100Gi”容量。



重要

测试显示，在 RHEL 中使用 NFS 服务器作为核心服务的存储后端可能会出现问题。这包括 OpenShift Container Registry 和 Quay，Prometheus 用于监控存储，以及 Elasticsearch 用于日志存储。因此，不推荐使用 RHEL NFS 作为 PV 后端用于核心服务。

市场上的其他 NFS 实现可能没有这些问题。如需了解更多与此问题相关的信息，请联络相关的 NFS 厂商。

流程

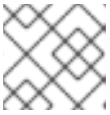
1. 为了配置 registry 使用存储，需要修改 **configs.imageregistry/cluster** 资源中的 **spec.storage.pvc** 。

注意

使用共享存储时，请查看您的安全设置以防止被外部访问。

2. 验证您没有 registry pod:

```
$ oc get pod -n openshift-image-registry
```



注意

如果存储类型为 **emptyDIR**，则副本数不能超过 **1**。

3. 检查 registry 配置：

```
$ oc edit configs.imageregistry.operator.openshift.io
```

输出示例

```
storage:
  pvc:
    claim: 1
```

1 将 **claim** 字段留空以允许自动创建一个 **image-registry-storage** PVC。

4. 检查 **clusteroperator** 的状态：

```
$ oc get clusteroperator image-registry
```

11.2.13.2.2. 为 VMware vSphere 配置块 registry 存储

在作为集群管理员升级时，要允许镜像 registry 使用块存储类型，如 vSphere Virtual Machine Disk（VMDK），您可以使用 **Recreate** rollout 策略。

重要

支持块存储卷，但不建议将其用于生产环境中的镜像 registry。在块存储上配置 registry 的安装不具有高可用性，因为 registry 无法拥有多个副本。

流程

1. 要将镜像 registry 存储设置为块存储类型，对 registry 进行补丁，使其使用 **Recreate** rollout 策略，且仅使用 **1** 个副本运行：

```
$ oc patch config.imageregistry.operator.openshift.io/cluster --type=merge -p '{"spec":
{"rolloutStrategy":"Recreate","replicas":1}}'
```

2. 为块存储设备置备 PV，并为该卷创建 PVC。请求的块卷使用 ReadWriteOnce（RWO）访问模式。

   a. 创建包含以下内容的 **pvc.yaml** 文件以定义 VMware vSphere **PersistentVolumeClaim**：

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: image-registry-storage 1
  namespace: openshift-image-registry 2
spec:
  accessModes:
  - ReadWriteOnce 3
  resources:
    requests:
      storage: 100Gi 4
```

   1. 代表 **PersistentVolumeClaim** 对象的唯一名称。
   2. **PersistentVolumeClaim** 对象的命名空间，即 **openshift-image-registry**。
   3. 持久性卷声明的访问模式。使用 **ReadWriteOnce** 时，单个节点可以通过读写权限挂载这个卷。
   4. 持久性卷声明的大小。

   b. 从文件创建 **PersistentVolumeClaim** 对象：

```
$ oc create -f pvc.yaml -n openshift-image-registry
```

3. 编辑 registry 配置，使其可以正确引用 PVC:

```
$ oc edit config.imageregistry.operator.openshift.io -o yaml
```

输出示例

```
storage:
  pvc:
    claim: 1
```

1. 通过创建自定义 PVC，您可以将 **claim** 字段留空以用于默认自动创建 **image-registry-storage** PVC。

有关配置 registry 存储以便引用正确的 PVC 的说明，请参阅为 vSphere 配置 registry。

11.2.14. 备份 VMware vSphere 卷

OpenShift Container Platform 将新卷作为独立持久性磁盘置备，以便在集群中的任何节点上自由附加和分离卷。因此，无法备份使用快照的卷，也无法从快照中恢复卷。如需更多信息，请参阅 快照限制。

流程

要创建持久性卷的备份：

1. 停止使用持久性卷的应用程序。
2. 克隆持久性卷。
3. 重启应用程序。
4. 创建克隆的卷的备份。
5. 删除克隆的卷。

11.2.15. OpenShift Container Platform 的 Telemetry 访问

在 OpenShift Container Platform 4.6 中，默认运行的 Telemetry 服务提供有关集群健康状况和成功更新的指标，需要访问互联网。如果您的集群连接到互联网，Telemetry 会自动运行，而且集群会注册到 OpenShift Cluster Manager。

确认 OpenShift Cluster Manager 清单正确后，可以由 Telemetry 自动维护，也可以使用 OpenShift Cluster Manager 手动维护，使用订阅监控来跟踪帐户或多集群级别的 OpenShift Container Platform 订阅。

其他资源

- 有关 Telemetry 服务的更多信息，请参阅关于远程健康监控。

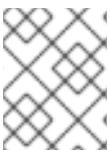
11.2.16. 后续步骤

- 自定义集群。
- 如果需要，您可以选择不使用远程健康报告。
- 设置 registry 并配置 registry 存储。

11.3. 使用网络自定义在 VSPHERE 上安装集群

在 OpenShift Container Platform 版本 4.6 中，您可以使用安装程序置备的基础架构和自定义的网络配置选项在 VMware vSphere 实例上安装集群。通过自定义网络配置，您的集群可以与环境中现有的 IP 地址分配共存，并与现有的 MTU 和 VXLAN 配置集成。要自定义安装，请在安装集群前修改 **install-config.yaml** 文件中的参数。

大部分网络配置参数必须在安装过程中设置，只有 **kubeProxy** 配置参数可以在运行的集群中修改。

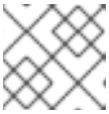


注意

OpenShift Container Platform 支持将集群部署到单个 VMware vCenter 中。不支持在多个 vCenter 上使用机器/机器集部署集群。

11.3.1. 先决条件

- 为集群置备持久性存储。若要部署私有镜像 registry，您的存储必须提供 **ReadWriteMany** 访问模式。
- 查看有关 OpenShift Container Platform 安装和更新流程的详细信息。
- OpenShift Container Platform 安装程序需要访问 vCenter 和 ESXi 主机上的端口 443。您确认可以访问端口 443。
- 如果您使用防火墙，请与管理员一起确认可以访问端口 443。control plane 节点必须能够通过端口 443 访问 vCenter 和 ESXi 主机，才能成功安装。
- 如果使用防火墙，则必须将其配置为允许集群需要访问的站点。



注意

如果您要配置代理，请务必也要查看此站点列表。

11.3.2. OpenShift Container Platform 的互联网访问

在 OpenShift Container Platform 4.6 中，您需要访问互联网来安装集群。

您必须具有以下互联网访问权限：

- 访问 OpenShift Cluster Manager 以下载安装程序并执行订阅管理。如果集群可以访问互联网，并且没有禁用 Telemetry，该服务会自动授权您的集群。
- 访问 Quay.io，以获取安装集群所需的软件包。
- 获取执行集群更新所需的软件包。

重要

如果您的集群无法直接访问互联网，则可以在置备的某些类基础架构上执行受限网络安装。在此过程中，您要下载所需的内容，并使用它在镜像 registry（mirror registry） 中填充安装集群并生成安装程序所需的软件包。对于某些安装类型，集群要安装到的环境不需要访问互联网。在更新集群之前，要更新 registry 镜像系统中的内容。

11.3.3. VMware vSphere 基础架构要求

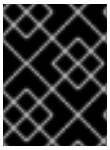
您必须在满足您使用的组件要求的 VMware vSphere 版本 6 或 7 实例上安装 OpenShift Container Platform 集群。

表 11.16. VMware 组件支持的最低 vSphere 版本

组件	最低支持版本	描述
虚拟机监控程序	vSphere 6.5 及之后的版本 13	此版本是 Red Hat Enterprise Linux CoreOS(RHCOS)支持的最低版本。请查看 Red Hat Enterprise Linux 8 支持的管理程序列表。

组件	最低支持版本	描述
使用 in-tree 驱动程序存储	vSphere 6.5 及之后的版本	此插件使用 OpenShift Container Platform 中包含的 vSphere 的树内存储驱动程序创建 vSphere 存储。
可选： Networking (NSX-T)	vSphere 6.5U3 或 vSphere 6.7U2 及之后的版本	OpenShift Container Platform 需要 vSphere 6.5U3 或 vSphere 6.7U2+。VMware 的 NSX Container Plug-in（NCP）3.0.2 使用 OpenShift Container Platform 4.6 和 NSX-T 3.x+ 认证。

如果您使用 vSphere 版本 6.5 实例，请在安装 OpenShift Container Platform 前考虑升级到 6.7U3 或 7.0。



重要

您必须确保在安装 OpenShift Container Platform 前同步 ESXi 主机上的时间。请参阅 VMware 文档中的编辑主机时间配置。

11.3.4. 网络连接要求

您必须配置机器之间的网络连接，以允许 OpenShift Container Platform 集群组件进行通信。

查看有关所需网络端口的以下详细信息。

表 11.17. 用于全机器到所有机器通信的端口

协议	端口	描述
ICMP	N/A	网络可访问性测试
TCP	**1936**	指标
	9000-9999	主机级别的服务，包括端口 **9100-9101** 上的节点导出器和端口 **9099** 上的 Cluster Version Operator。
	10250-10259	Kubernetes 保留的默认端口
	10256	openshift-sdn
UDP	**4789**	虚拟可扩展 LAN(VXLAN)
	6081	Geneve
	9000-9999	主机级别的服务，包括端口 **9100-9101** 上的节点导出器。

协议	端口	描述
	500	IPsec IKE 数据包
	4500	IPsec NAT-T 数据包
TCP/UDP	**30000-32767**	Kubernetes 节点端口
ESP	N/A	IPsec Encapsulating Security Payload(ESP)

表 11.18. 用于所有机器控制平面通信的端口

协议	端口	描述
TCP	**6443**	Kubernetes API

表 11.19. control plane 机器用于 control plane 机器通信的端口

协议	端口	描述
TCP	**2379-2380**	etcd 服务器和对等端口

11.3.5. vCenter 要求

在使用安装程序置备的基础架构的 vCenter 上安装 OpenShift Container Platform 集群前，您必须准备自己的环境。

所需的 vCenter 帐户权限
要在 vCenter 中安装 OpenShift Container Platform 集群，安装程序需要一个具有特权的帐户来读取和创建所需资源。使用具有全局管理特权的帐户是访问所有必要权限的最简单方式。

如果无法使用具有全局管理特权的帐户，您必须创建角色来授予 OpenShift Container Platform 集群安装所需的权限。虽然大多数权限始终是必需的，但是一些权限只有在计划安装程序需要在您的 vCenter 实例中置备一个包含 OpenShift Container Platform 集群的文件夹时（这是默认行为）才需要。您必须为指定对象创建或修改 vSphere 角色，才能授予所需的权限。

如果安装程序创建 vSphere 虚拟机文件夹，则需要额外的角色。

例 11.5. 安装所需的角色和权限

角色的 vSphere 对象	何时需要	所需的权限

角色的 vSphere 对象	何时需要	所需的权限
vSphere vCenter	Always	**Cns.Searchable** **InventoryService.Tagging.AttachTag** **InventoryService.Tagging.CreateCategory** **InventoryService.Tagging.CreateTag** **InventoryService.Tagging.DeleteCategory** **InventoryService.Tagging.DeleteTag** **InventoryService.Tagging.EditCategory** **InventoryService.Tagging.EditTag** **Sessions.ValidateSession** **StorageProfile.View**
vSphere vCenter Cluster	Always	**Host.Config.Storage** **Resource.AssignVMToPool** **VApp.AssignResourcePool** **VApp.Import** **VirtualMachine.Config.AddNewDisk**
vSphere Datastore	Always	**Datastore.AllocateSpace** **Datastore.Browse** **Datastore.FileManagement**
vSphere 端口组	Always	**Network.Assign**

角色的 vSphere 对象	何时需要	所需的权限
虚拟机文件夹	Always	**Resource.AssignVMToPool VApp.Import VirtualMachine.Config.Add ExistingDisk VirtualMachine.Config.Add NewDisk VirtualMachine.Config.Add RemoveDevice VirtualMachine.Config.Adva ncedConfig VirtualMachine.Config.Anno tation VirtualMachine.Config.CPU Count VirtualMachine.Config.Disk Extend VirtualMachine.Config.Disk Lease VirtualMachine.Config.Edit Device VirtualMachine.Config.Mem ory VirtualMachine.Config.Rem oveDisk VirtualMachine.Config.Rena me VirtualMachine.Config.Rese tGuestInfo VirtualMachine.Config.Reso urce VirtualMachine.Config.Setti ngs VirtualMachine.Config.Upgr adeVirtualHardware VirtualMachine.Interact.Gue stControl VirtualMachine.Interact.Pow erOff VirtualMachine.Interact.Pow erOn VirtualMachine.Interact.Res et VirtualMachine.Inventory.Cr eate VirtualMachine.Inventory.Cr eateFromExisting VirtualMachine.Inventory.D elete VirtualMachine.Provisionin g.Clone**

角色的 vSphere 对象	何时需要	所需的权限
vSphere vCenter Datacenter	如果安装程序创建虚拟机文件夹	**Resource.AssignVMToPool VApp.Import VirtualMachine.Config.Add ExistingDisk VirtualMachine.Config.Add NewDisk VirtualMachine.Config.Add RemoveDevice VirtualMachine.Config.Adva ncedConfig VirtualMachine.Config.Anno tation VirtualMachine.Config.CPU Count VirtualMachine.Config.Disk Extend VirtualMachine.Config.Disk Lease VirtualMachine.Config.Edit Device VirtualMachine.Config.Mem ory VirtualMachine.Config.Rem oveDisk VirtualMachine.Config.Rena me VirtualMachine.Config.Rese tGuestInfo VirtualMachine.Config.Reso urce VirtualMachine.Config.Setti ngs VirtualMachine.Config.Upgr adeVirtualHardware VirtualMachine.Interact.Gue stControl VirtualMachine.Interact.Pow erOff VirtualMachine.Interact.Pow erOn VirtualMachine.Interact.Res et VirtualMachine.Inventory.Cr eate VirtualMachine.Inventory.Cr eateFromExisting VirtualMachine.Inventory.D elete VirtualMachine.Provisionin g.Clone Folder.Create Folder.Delete**

此外，用户需要一些 **ReadOnly** 权限，某些角色需要权限来提升对子对象的权限。这些设置会根据您是否将集群安装到现有文件夹而有所不同。

例 11.6. 所需的权限和传播设置

vSphere 对象	文件夹类型	传播到子对象	所需的权限
vSphere vCenter	Always	False	列出所需的权限
vSphere vCenter Datacenter	现有文件夹	False	**ReadOnly** 权限
	安装程序创建文件夹	True	列出所需的权限
vSphere vCenter Cluster	Always	True	列出所需的权限
vSphere vCenter Datastore	Always	False	列出所需的权限
vSphere Switch	Always	False	**ReadOnly** 权限
vSphere 端口组	Always	False	列出所需的权限
vSphere vCenter Virtual Machine Folder	现有文件夹	True	列出所需的权限

有关只使用所需权限创建帐户的更多信息，请参阅 vSphere 文档中的 vSphere 权限和用户管理任务。

将 OpenShift Container Platform 与 vMotion 搭配使用

如果要在 vSphere 环境中使用 vMotion，请在安装 OpenShift Container Platform 集群前考虑以下内容。

- OpenShift Container Platform 通常支持仅用于计算的 vMotion。使用 Storage vMotion 可能会导致问题且不被支持。
  为了帮助确保计算和 control plane 节点的正常运行时间，建议您遵循 VMware 最佳实践进行 vMotion。还建议使用 VMware 反关联性规则来改进 OpenShift Container Platform 在维护或硬件问题期间的可用性。

  有关 vMotion 和 anti-affinity 规则的更多信息，请参阅 VMware vSphere 文档 了解 vMotion 网络要求和虚拟机反关联性规则。

- 如果您在 pod 中使用 vSphere 卷，请手动或通过 Storage vMotion 在数据存储间迁移虚拟机，从而导致 OpenShift Container Platform 持久性卷(PV)对象中的无效引用。这些引用可防止受影响的 pod 启动，并可能导致数据丢失。

- 同样，OpenShift Container Platform 不支持在数据存储间有选择地迁移 VMDK、使用数据存储集群进行虚拟机置备、动态或静态置备 PV，或使用作为数据存储集群一部分的数据存储进行 PV 的动态或静态置备。

集群资源

当部署使用安装程序置备的基础架构的 OpenShift Container Platform 集群时，安装程序必须能够在 vCenter 实例中创建多个资源。

标准 OpenShift Container Platform 安装会创建以下 vCenter 资源：

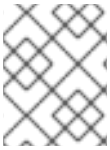
- 1 个文件夹
- 1 标签（Tag）类别
- 1 个标签（Tag）
- 虚拟机:
  - 1 个模板
  - 1 个临时 bootstrap 节点
  - 3 个 control plane 节点
  - 3 个计算机器

虽然这些资源使用了 856 GB 存储，但 bootstrap 节点会在集群安装过程中被销毁。使用标准集群至少需要 800 GB 存储。

如果部署了更多计算机器，OpenShift Container Platform 集群将使用更多存储。

集群的限制
可用资源因集群而异。vCenter 中可能的集群数量主要受可用存储空间以及对所需资源数量的限制。确保考虑集群创建的 vCenter 资源的限制和部署集群所需的资源，如 IP 地址和网络。

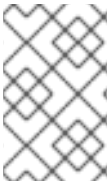
网络要求
网络必须使用 DHCP，并确保 DHCP 服务器被配置为为集群机器提供持久的 IP 地址。



注意

在安装开始前，持久性 IP 地址不可用。分配 DHCP 范围后，在安装后使用持久 IP 地址手动替换分配。

另外，在安装 OpenShift Container Platform 集群前，必须创建以下网络资源：



注意

建议集群中的每个 OpenShift Container Platform 节点都可以访问可通过 DHCP 发现的网络时间协议（NTP）服务器。没有 NTP 服务器也可安装。但是，异步服务器时钟将导致错误，NTP 服务器会阻止。

所需的 IP 地址
安装程序置备的 vSphere 安装需要这些静态 IP 地址：

- API 地址用于访问集群 API。
- Ingress 地址用于集群入口流量。
- 当将集群从版本 4.5 升级到 4.6 时，会使用 control plane 节点地址。

安装 OpenShift Container Platform 集群时，必须向安装程序提供这些 IP 地址。

DNS 记录

您必须在正确的 DNS 服务器中为托管 OpenShift Container Platform 集群的 vCenter 实例创建两个静态 IP 地址的 DNS 记录。在每个记录中，**<cluster_name>** 是集群名称，**<base_domain>** 是您在安装集群时指定的集群基域。完整的 DNS 记录采用如下格式: **<component>.<cluster_name>.<base_domain>.**。

表 11.20. 所需的 DNS 记录

组件	记录	描述
API VIP	**api.<cluster_name>.<base_domain>.**	此 DNS A/AAAA 或 CNAME 记录必须指向 control plane 机器的负载均衡器。此记录必须能由集群外的客户端和集群内的所有节点解析。
Ingress VIP	***.apps.<cluster_name>.<base_domain>.**	通配符 DNS A/AAAA 或 CNAME 记录，指向以运行入口路由器 Pod 的机器（默认为 worker 节点）为目标的负载均衡器。此记录必须能由集群外的客户端和集群内的所有节点解析。

11.3.6. 生成 SSH 私钥并将其添加到代理中

如果要在集群上执行安装调试或灾难恢复，则必须为 **ssh-agent** 和安装程序提供 SSH 密钥。您可以使用此密钥访问公共集群中的 bootstrap 机器来排除安装问题。

注意

在生产环境中，您需要进行灾难恢复和调试。

您可以使用此密钥以 **core** 用户身份通过 SSH 连接到 master 节点。在部署集群时，此密钥会添加到 **core** 用户的 **~/.ssh/authorized_keys** 列表中。

注意

您必须使用一个本地密钥，而不要使用在特定平台上配置的密钥，如 AWS 密钥对。

流程

1. 如果还没有为计算机上免密码身份验证而配置的 SSH 密钥，请创建一个。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ ssh-keygen -t ed25519 -N '' \
  -f <path>/<file_name> 1
```

1 指定新 SSH 密钥的路径和文件名，如 **~/.ssh/id_rsa**。如果您已有密钥对，请确保您的公钥位于 **~/.ssh** 目录中。

运行此命令会在指定的位置生成不需要密码的 SSH 密钥。

注意

如果您计划在 **x86_64** 架构中安装使用 FIPS 验证的/Modules in Process 加密库的 OpenShift Container Platform 集群，不要创建使用 **ed25519** 算法的密钥。反之，创建一个使用 **rsa** 或 **ecdsa** 算法的密钥。

2. 作为后台任务启动 **ssh-agent** 进程：

```
$ eval "$(ssh-agent -s)"
```

输出示例

```
Agent pid 31874
```

注意

如果您的集群采用 FIPS 模式，则只使用 FIPS 兼容算法来生成 SSH 密钥。密钥必须是 RSA 或 ECDSA。

3. 将 SSH 私钥添加到 **ssh-agent**：

```
$ ssh-add <path>/<file_name> (1)
```

输出示例

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

(1) 指定 SSH 私钥的路径和文件名，如 **~/.ssh/id_rsa**

后续步骤

- 在安装 OpenShift Container Platform 时，为安装程序提供 SSH 公钥。

11.3.7. 获取安装程序

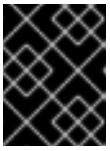
在安装 OpenShift Container Platform 之前，将安装文件下载到本地计算机上。

先决条件

- 运行 Linux 或 macOS 的计算机，本地磁盘空间为 500 MB

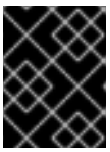
流程

1. 访问 OpenShift Cluster Manager 站点的 Infrastructure Provider 页面。如果您有红帽帐号，请使用自己的凭证登录。如果没有，请创建一个帐户。
2. 选择您的基础架构供应商。
3. 进入适用于您的安装类型的页面，下载您的操作系统的安装程序，并将文件放在要保存安装配置文件的目录中。。



重要

安装程序会在用来安装集群的计算机上创建若干文件。在完成集群安装后，您必须保留安装程序和安装程序所创建的文件。这两个文件都需要删除集群。



重要

删除安装程序创建的文件不会删除您的集群，即使集群在安装过程中失败也是如此。要删除集群，为特定云供应商完成 OpenShift Container Platform 卸载流程。

4. 提取安装程序。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ tar xvf openshift-install-linux.tar.gz
```

5. 从 Red Hat OpenShift Cluster Manager 下载安装 pull secret 。通过此 pull secret，您可以进行所含授权机构提供的服务的身份验证，这些服务包括为 OpenShift Container Platform 组件提供容器镜像的 Quay.io。

11.3.8. 在您的系统信任中添加 vCenter root CA 证书

由于安装程序需要访问 vCenter 的 API，所以必须在安装 OpenShift Container Platform 集群前将 vCenter 的可信 root CA 证书添加到系统信任中。

流程

1. 在 vCenter 主页中下载 vCenter 的 root CA 证书。在 vSphere Web Services SDK 部分点击 **Download trusted root CA certificates**。**<vCenter>/certs/download.zip** 文件下载。

2. 提取包含 vCenter root CA 证书的压缩文件。压缩文件的内容类似以下文件结构：

```
certs
├── lin
│   ├── 108f4d17.0
│   ├── 108f4d17.r1
│   ├── 7e757f6a.0
│   ├── 8e4f8471.0
│   └── 8e4f8471.r0
├── mac
│   ├── 108f4d17.0
│   ├── 108f4d17.r1
│   ├── 7e757f6a.0
│   ├── 8e4f8471.0
│   └── 8e4f8471.r0
└── win
    ├── 108f4d17.0.crt
    ├── 108f4d17.r1.crl
    ├── 7e757f6a.0.crt
    ├── 8e4f8471.0.crt
    └── 8e4f8471.r0.crl

3 directories, 15 files
```

3. 将您的操作系统的文件添加到系统信任中。例如，在 Fedora 操作系统上运行以下命令：

```
# cp certs/lin/* /etc/pki/ca-trust/source/anchors
```

4. 更新您的系统信任关系。例如，在 Fedora 操作系统上运行以下命令：

```
# update-ca-trust extract
```

11.3.9. 创建安装配置文件

您可以自定义在 VMware vSphere 上安装的 OpenShift Container Platform 集群。

先决条件

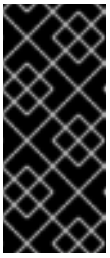
- 获取 OpenShift Container Platform 安装程序以及集群的 pull secret。

流程

1. 创建 **install-config.yaml** 文件。
   a. 更改到包含安装程序的目录，再运行以下命令：

```
$ ./openshift-install create install-config --dir <installation_directory> 1
```

   1 对于 **<installation_directory>**，请指定用于保存安装程序所创建的文件的目录名称。



重要

指定一个空目录。一些安装信息，如 bootstrap X.509 证书，有较短的过期间隔，因此不要重复使用安装目录。如果要重复使用另一个集群安装中的个别文件，可以将其复制到您的目录中。但是，一些安装数据的文件名可能会在发行版本之间有所改变。从 OpenShift Container Platform 老版本中复制安装文件时要格外小心。

   b. 在提示符处，提供您的云的配置详情：
      i. 可选：选择用来访问集群机器的 SSH 密钥。

注意

对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 **ssh-agent** 进程使用的 SSH 密钥。

      ii. 选择 **vsphere** 作为目标平台。
      iii. 指定 vCenter 实例的名称。
      iv. 指定创建集群所需的权限的 vCenter 帐户的用户名和密码。
      安装程序连接到您的 vCenter 实例。
      v. 选择要连接的 vCenter 实例中的数据中心。
      vi. 选择要使用的默认 vCenter 数据存储。

vii. 选择要在其中安装 vCenter 集群的 OpenShift Container Platform 集群。安装程序使用 vSphere 集群的 root 资源池作为默认资源池。

viii. 选择包含您配置的虚拟 IP 地址和 DNS 记录的 vCenter 实例中的网络。

ix. 输入您为 control plane API 访问配置的虚拟 IP 地址。

x. 输入您为集群入口配置的虚拟 IP 地址。

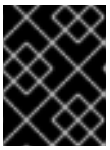
xi. 输入基域。这个基域必须与您配置的 DNS 记录中使用的域相同。

xii. 为集群输入一个描述性名称。集群名称必须与您配置的 DNS 记录中使用的相同。

xiii. 粘贴 Red Hat OpenShift Cluster Manager 中的 pull secret 。

2. 修改 **install-config.yaml** 文件。您可以在**安装配置参数**部分中找到有关可用参数的更多信息。

3. 备份 **install-config.yaml** 文件，以便用于安装多个集群。



重要

install-config.yaml 文件会在安装过程中消耗掉。如果要重复使用此文件，必须现在备份。

11.3.9.1. 安装配置参数

在部署 OpenShift Container Platform 集群前，您可以提供参数值，以描述托管集群的云平台的帐户并选择性地自定义集群平台。在创建 **install-config.yaml** 安装配置文件时，您可以通过命令行来提供所需的参数的值。如果要自定义集群，可以修改 **install-config.yaml** 文件来提供关于平台的更多信息。

注意

安装之后，您无法修改 **install-config.yaml** 文件中的这些参数。

重要

openshift-install 命令不验证参数的字段名称。如果指定了不正确的名称，则不会创建相关的文件或对象，且不会报告错误。确保所有指定的参数的字段名称都正确。

11.3.9.1.1. 所需的配置参数

下表描述了所需的安装配置参数：

表 11.21. 所需的参数

参数	描述	值
apiVersion	**install-config.yaml** 内容的 API 版本。当前版本是 **v1**。安装程序还可能支持旧的 API 版本。	字符串

参数	描述	值
baseDomain	云供应商的基域。此基础域用于创建到 OpenShift Container Platform 集群组件的路由。集群的完整 DNS 名称是 **baseDomain** 和 **metadata.name** 参数值的组合，其格式为 **<metadata.name>.<baseDomain>**。	完全限定域名或子域名，如 **example.com**。
metadata	Kubernetes 资源 **ObjectMeta**，其中只消耗 **name** 参数。	对象
metadata.name	集群的名称。集群的 DNS 记录是 **{{.metadata.name}}.{{.baseDomain}}** 的子域。	小写字母和连字符(**-**)的字符串，如 **dev**。
platform	执行安装的具体平台配置：**aws**、**baremetal**、**azure**、**openstack**、**ovirt**、**vsphere**。有关 **platform.<platform>** 参数的额外信息，请参考下表来了解您的具体平台。	对象

pullSecret	从 Red Hat OpenShift Cluster Manager 获取 pull secret，验证从 Quay.io 等服务中下载 OpenShift Container Platform 组件的容器镜像。	<pre>{
		  "auths":{
		  "cloud.openshift.com":{
		  "auth":"b3Blb=",
		  "email":"you@example.com"
		  },
		  "quay.io":{
		  "auth":"b3Blb=",
		  "email":"you@example.com"
		  }
		  }
		}</pre>

11.3.9.1.2. 网络配置参数

您可以根据现有网络基础架构的要求自定义安装配置。例如，您可以扩展集群网络的 IP 地址块，或者提供不同于默认值的不同 IP 地址块。

只支持 IPv4 地址。

表 11.22. 网络参数

参数	描述	值
networking	集群网络的配置。	对象 **注意** 您不能在安装后修改 **networking** 对象指定的参数。
networking.networkType	要安装的集群网络供应商 Container Network Interface（CNI）插件。	**OpenShiftSDN** 或 **OVNKubernetes**。默认值为 **OpenShiftSDN**。
networking.clusterNetwork	pod 的 IP 地址块。 默认值为 **10.128.0.0/14**，主机前缀为 **/23**。 如果您指定多个 IP 地址块，则块不得互相重叠。	一个对象数组。例如： networking: clusterNetwork: - cidr: 10.128.0.0/14 hostPrefix: 23
networking.clusterNetwork.cidr	使用 **networking.clusterNetwork** 时需要此项。IP 地址块。 一个 IPv4 网络。	使用 CIDR 形式的 IP 地址块。IPv4 块的前缀长度介于 **0** 到 **32** 之间。
networking.clusterNetwork.hostPrefix	分配给每个单独节点的子网前缀长度。例如，如果 **hostPrefix** 设为 **23**，则每个节点从所给的 **cidr** 中分配一个 **/23** 子网。**hostPrefix** 值 **23** 提供 510（2^(32 - 23)- 2）个 pod IP 地址。	子网前缀。 默认值为 **23**。
networking.serviceNetwork	服务的 IP 地址块。默认值为 **172.30.0.0/16**。 OpenShift SDN 和 OVN-Kubernetes 网络供应商只支持服务网络的一个 IP 地址块。	CIDR 格式具有 IP 地址块的数组。例如： networking: serviceNetwork: - 172.30.0.0/16
networking.machineNetwork	机器的 IP 地址块。 如果您指定多个 IP 地址块，则块不得互相重叠。	一个对象数组。例如： networking: machineNetwork: - cidr: 10.0.0.0/16

参数	描述	值
networking.machineNetwork.cidr	使用 **networking.machineNetwork** 时需要。IP 地址块。libvirt 以外的所有平台的默认值为 **10.0.0.0/16**。对于 libvirt，默认值为 **192.168.126.0/24**。	CIDR 表示法中的 IP 网络块。 例如： **10.0.0.0/16**。  **注意** 将 **networking.machineNetwork** 设置为与首选 NIC 所在的 CIDR 匹配。

11.3.9.1.3. 可选配置参数

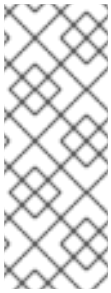
下表描述了可选安装配置参数：

表 11.23. 可选参数

参数	描述	值
additionalTrustBundle	添加到节点可信证书存储中的 PEM 编码 X.509 证书捆绑包。配置了代理时，也可以使用这个信任捆绑包。	字符串
compute	组成计算节点的机器的配置。	machine-pool 对象的数组。详情请查看以下"Machine-pool"表。
compute.architecture	决定池中机器的指令集合架构。目前不支持异构集群，因此所有池都必须指定相同的架构。有效值为 **amd64** （默认值）。	字符串
compute.hyperthreading	是否在计算机器上启用或禁用并发多线程或**超线程**。默认情况下，启用并发多线程以提高机器内核的性能。  **重要** 如果禁用并发多线程，请确保在容量规划时考虑到机器性能可能会显著降低的问题。	**Enabled** 或 **Disabled**
compute.name	使用 **compute** 时需要此值。机器池的名称。	**worker**

参数	描述	值
compute.platform	使用 **compute** 时需要此值。使用此参数指定托管 worker 机器的云供应商。此参数值必须与 **controlPlane.platform** 参数值匹配。	**aws**、**azure**、**gcp**、**openstack**、**ovirt**、**vsphere** 或 **{}**
compute.replicas	要置备的计算机器数量，也称为 worker 机器。	大于或等于 **2** 的正整数。默认值为 **3**。
controlPlane	组成 control plane 的机器的配置。	**MachinePool** 对象的数组。详情请查看以下"Machine-pool"表。
controlPlane.architecture	决定池中机器的指令集合架构。目前不支持异构集群，因此所有池都必须指定相同的架构。有效值为 **amd64** （默认值）。	字符串
controlPlane.hyperthreading	是否在 control plane 机器上启用或禁用并发多线程或**超线程**。默认情况下，启用并发多线程以提高机器内核的性能。  **重要** 如果禁用并发多线程，请确保在容量规划时考虑到机器性能可能会显著降低的问题。	**Enabled** 或 **Disabled**
controlPlane.name	使用 **controlPlane** 时需要。机器池的名称。	**master**
controlPlane.platform	使用 **controlPlane** 时需要。使用此参数指定托管 control plane 机器的云供应商。此参数值必须与 **compute.platform** 参数值匹配。	**aws**、**azure**、**gcp**、**openstack**、**ovirt**、**vsphere** 或 **{}**
controlPlane.replicas	要置备的 control plane 机器数量。	唯一支持的值是 **3**，它是默认值。

参数	描述	值
credentialsMode	Cloud Credential Operator（CCO）模式。如果没有指定任何模式，CCO 会动态地尝试决定提供的凭证的功能，在支持多个模式的平台上使用 mint 模式。  **注意** 不是所有 CCO 模式都支持所有云供应商。如需有关 CCO 模式的更多信息，请参阅*Red Hat Operator 参考指南*内容中的 *Cloud Credential Operator* 条目。	**Mint**、**Passthrough**、**Manual** 或空字符串(**""**)。
fips	启用或禁用 FIPS 模式。默认为 **false**（禁用）。如果启用了 FIPS 模式，运行 OpenShift Container Platform 的 Red Hat Enterprise Linux CoreOS (RHCOS) 机器会绕过默认的 Kubernetes 加密套件，并使用由 RHCOS 提供的加密模块。  **重要** 只有在 **x86_64** 架构中的 OpenShift Container Platform 部署支持 FIPS 验证的/Modules in Process 加密库。  **注意** 如果使用 Azure File 存储，则无法启用 FIPS 模式。	**false** 或 **true**
imageContentSources	release-image 内容的源和仓库。	对象数组。包括一个 **source** 以及可选的 **mirrors**，如下表所示。
imageContentSources.source	使用 **imageContentSources** 时需要。指定用户在镜像拉取规格中引用的仓库。	字符串
imageContentSources.mirrors	指定可能还包含同一镜像的一个或多个仓库。	字符串数组

参数	描述	值
publish	如何发布或公开集群的面向用户的端点，如 Kubernetes API、OpenShift 路由。	**Internal** 或 **External**。默认值为 **External**。 在非云平台上不支持将此字段设置为 **Internal**。  **重要** 如果将字段的值设为 **Internal**，集群将无法运行。如需更多信息，请参阅 BZ#1953035。
sshKey	用于验证集群机器访问的 SSH 密钥或密钥。  **注意** 对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 **ssh-agent** 进程使用的 SSH 密钥。	一个或多个密钥。例如： sshKey: <key1> <key2> <key3>

11.3.9.1.4. 其他 VMware vSphere 配置参数

下表描述了其他 VMware vSphere 配置参数：

表 11.24. 其他 VMware vSphere 集群参数

参数	描述	值
platform.vsphere.vCenter	vCenter 服务器的完全限定主机名或 IP 地址。	字符串
platform.vsphere.username	用于连接 vCenter 实例的用户名。此用户必须至少具有 vSphere 中 静态或动态持久性卷置备 所需的角色和权限。	字符串
platform.vsphere.password	vCenter 用户名的密码。	字符串

参数	描述	值
platform.vsphere.datacenter	要在 vCenter 实例中使用的数据中心的名称。	字符串
platform.vsphere.defaultDatastore	用于置备卷的默认数据存储名称。	字符串
platform.vsphere.folder	*可选*。安装程序创建虚拟机的现有文件夹的绝对路径。如果没有提供这个值，安装程序会创建一个文件夹，它的名称是数据中心虚拟机文件夹中的基础架构 ID。	字符串，如 **/<datacenter_name>/vm/<folder_name>/<subfolder_name>**。
platform.vsphere.network	包含您配置的虚拟 IP 地址和 DNS 记录的 vCenter 实例中的网络。	字符串
platform.vsphere.cluster	在其中安装 OpenShift Container Platform 集群的 vCenter 集群。	字符串
platform.vsphere.apiVIP	为 control plane API 访问配置的虚拟 IP（VIP）地址。	IP 地址，如 **128.0.0.1**。
platform.vsphere.ingressVIP	为集群入口配置的虚拟 IP（VIP）地址。	IP 地址，如 **128.0.0.1**。

11.3.9.1.5. 可选的 VMware vSphere 机器池配置参数

下表描述了可选的 VMware vSphere 机器池配置参数：

表 11.25. 可选的 VMware vSphere 机器池参数

参数	描述	值
platform.vsphere.clusterOSImage	安装程序从中下载 RHCOS 镜像的位置。您必须设置此参数以便在受限网络中执行安装。	HTTP 或 HTTPS URL，可选使用 SHA-256 checksum。例如：**https://mirror.openshift.com/images/rhcos-<version>-vmware.<architecture>.ova**。
platform.vsphere.osDisk.diskSizeGB	以 GB 为单位的磁盘大小。	整数
platform.vsphere.cpus	分配虚拟机的虚拟处理器内核总数。	整数

参数	描述	值
platform.vsphere.coresPerSocket	虚拟机中每个插槽的内核数。虚拟机上的虚拟套接字数量为 **platform.vsphere.cpus/platform.vsphere.coresPerSocket**。默认值为 **1**。	整数
platform.vsphere.memoryMB	以 MB 为单位的虚拟机内存大小。	整数

11.3.9.2. 安装程序置备的 VMware vSphere 集群的 install-config.yaml 文件示例

您可以自定义 install-config.yaml 文件，以指定有关 OpenShift Container Platform 集群平台的更多信息，或修改所需参数的值。

```
apiVersion: v1
baseDomain: example.com (1)
compute: (2)
- hyperthreading: Enabled (3)
  name: worker
  replicas: 3
  platform:
    vsphere: (4)
      cpus: 2
      coresPerSocket: 2
      memoryMB: 8192
      osDisk:
        diskSizeGB: 120
controlPlane: (5)
  hyperthreading: Enabled (6)
  name: master
  replicas: 3
  platform:
    vsphere: (7)
      cpus: 4
      coresPerSocket: 2
      memoryMB: 16384
      osDisk:
        diskSizeGB: 120
metadata:
  name: cluster (8)
networking:
  clusterNetwork:
  - cidr: 10.128.0.0/14
    hostPrefix: 23
  machineNetwork:
  - cidr: 10.0.0.0/16
  networkType: OpenShiftSDN
  serviceNetwork:
  - 172.30.0.0/16
platform:
```

```
  vsphere:
    vcenter: your.vcenter.server
    username: username
    password: password
    datacenter: datacenter
    defaultDatastore: datastore
    folder: folder
    network: VM_Network
    cluster: vsphere_cluster_name 9
    apiVIP: api_vip
    ingressVIP: ingress_vip
fips: false
pullSecret: '{"auths": ...}'
sshKey: 'ssh-ed25519 AAAA...'
```

1 集群的基域。所有 DNS 记录都必须是这个基域的子域，并包含集群名称。

2 5 **controlPlane** 部分是一个单映射，但 **compute** 部分是一系列映射。为满足不同数据结构的要求，**compute** 部分的第一行必须以连字符 **-** 开头，**controlPlane** 部分的第一行则不可以连字符开头。只使用一个 control plane 池。

3 6 是否要启用或禁用并发多线程或**超线程**。默认情况下，启用并发多线程以提高机器内核的性能。您可以通过将参数值设为 **Disabled** 来禁用。如果您在某些集群机器上禁用并发多线程，则必须在所有集群机器上禁用。

重要

如果禁用并发多线程，请确保在容量规划时考虑到机器性能可能会显著降低的问题。如果您禁用并发多线程，则计算机必须至少使用 8 个 CPU 和 32GB RAM。

4 7 可选：为 compute 和 control plane 机器提供额外的机器池参数配置。

8 您在 DNS 记录中指定的集群名称。

9 要在其中安装 OpenShift Container Platform 集群的 vSphere 集群。安装程序使用 vSphere 集群的 root 资源池作为默认资源池。

11.3.9.3. 在安装过程中配置集群范围代理

生产环境可能会拒绝直接访问互联网，而是提供 HTTP 或 HTTPS 代理。您可以通过在 **install-config.yaml** 文件中配置代理设置，将新的 OpenShift Container Platform 集群配置为使用代理。

先决条件

- 您有一个现有的 **install-config.yaml** 文件。
- 您检查了集群需要访问的站点，并决定是否需要绕过代理。默认情况下代理所有集群出口流量，包括对托管云供应商 API 的调用。您需要将站点添加到 **Proxy** 对象的 **spec.noProxy** 字段来绕过代理。

注意

Proxy 对象 **status.noProxy** 字段使用安装配置中的 **networking.machineNetwork[].cidr**、**networking.clusterNetwork[].cidr** 和 **networking.serviceNetwork[]** 字段的值填充。

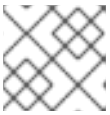
对于在 Amazon Web Services(AWS)、Google Cloud Platform(GCP)、Microsoft Azure 和 Red Hat OpenStack Platform(RHOSP)上安装, **Proxy** 对象 **status.noProxy** 字段也会使用实例元数据端点填充(**169.254.169.254**)。

流程

1. 编辑 **install-config.yaml** 文件并添加代理设置。例如：

```
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: example.com 3
additionalTrustBundle: | 4
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
...
```

1. 用于创建集群外 HTTP 连接的代理 URL。URL 必须是 **http**。
2. 用于创建集群外 HTTPS 连接的代理 URL。
3. 要排除在代理中的目标域名、IP 地址或其他网络 CIDR 的逗号分隔列表。在域前面加 **.** 来仅匹配子域。例如： **.y.com** 匹配 **x.y.com**，但不匹配 **y.com**。使用 * 绕过所有目的地的代理。您必须包含 vCenter 的 IP 地址以及用于其机器的 IP 范围。
4. 如果提供，安装程序会在 **openshift-config 命名空间中生成名为 user-ca- bundle** 的配置映射来保存额外的 CA 证书。如果您提供 **additionalTrustBundle** 和至少一个代理设置，则 **Proxy** 对象会被配置为引用 **trustedCA** 字段中的 **user-ca-bundle** 配置映射。然后，Cluster Network Operator 会创建一个 **trusted-ca-bundle** 配置映射，该配置映射将为 **trustedCA** 参数指定的内容与 RHCOS 信任捆绑包合并。 **additionalTrustBundle** 字段是必需的，除非代理的身份证书由来自 RHCOS 信任捆绑包的颁发机构签名。



注意

安装程序不支持代理的 **readinessEndpoints** 字段。

2. 保存该文件，并在安装 OpenShift Container Platform 时引用。

安装程序会创建一个名为 **cluster** 的集群范围代理，该代理使用提供的 **install-config.yaml** 文件中的代理设置。如果没有提供代理设置，仍然会创建一个 **cluster Proxy** 对象，但它会有一个空 **spec**。

注意

只支持名为 **cluster** 的 **Proxy** 对象，且无法创建额外的代理。

11.3.10. 网络配置阶段

当在安装前指定集群配置时，在安装过程中的几个阶段可以修改网络配置：

阶段 1

输入 **openshift-install create install-config** 命令后。在 **install-config.yaml** 文件中，您可以自定义以下与网络相关的字段：

- **networking.networkType**
- **networking.clusterNetwork**
- **networking.serviceNetwork**
- **networking.machineNetwork**
  有关这些字段的更多信息，请参阅"安装配置参数"。

注意

将 **networking.machineNetwork** 设置为与首选 NIC 所在的 CIDR 匹配。

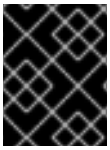
阶段 2

输入 **openshift-install create manifests** 命令后。如果必须指定高级网络配置，在这个阶段中，只能使用您要修改的字段来定义自定义的 Cluster Network Operator 清单。

在 2 阶段，您无法覆盖 **install-config.yaml** 文件中的 1 阶段中指定的值。但是，您可以在第 2 阶段进一步自定义集群网络供应商。

11.3.11. 指定高级网络配置

您可以通过为集群网络供应商指定额外的配置，使用高级配置自定义将集群整合到现有网络环境中。您只能在安装集群前指定高级网络配置。



重要

不支持修改安装程序创建的 OpenShift Container Platform 清单文件。支持应用您创建的清单文件，如以下流程所示。

先决条件

- 创建 **install-config.yaml** 文件并完成对其所做的任何修改。

流程

1. 进入包含安装程序的目录并创建清单：

```
$ ./openshift-install create manifests --dir <installation_directory>
```

其中：

<installation_directory>
指定包含集群的 **install-config.yaml** 文件的目录名称。

2. 在 **<installation_directory>/manifests/** 目录下，为高级网络配置创建一个名为 **cluster-network-03-config.yml** 的 stub 清单文件：

```
$ cat <<EOF > <installation_directory>/manifests/cluster-network-03-config.yml
apiVersion: operator.openshift.io/v1
kind: Network
metadata:
  name: cluster
spec:
EOF
```

其中：

<installation_directory>

指定包含集群的 **manifests/** 目录的目录名称。

3. 在编辑器中打开 **cluster-network-03-config.yml** 文件，并为集群指定高级网络配置，如下例所示：

为 OpenShift SDN 网络供应商指定不同的 VXLAN 端口

```
apiVersion: operator.openshift.io/v1
kind: Network
metadata:
  name: cluster
spec:
  defaultNetwork:
    openshiftSDNConfig:
      vxlanPort: 4800
```

4. 保存 **cluster-network-03-config.yml** 文件，再退出文本编辑器。

5. 可选：备份 **manifests/cluster-network-03-config.yml** 文件。创建集群时，安装程序会删除 **manifests/** 目录。

11.3.12. Cluster Network Operator 配置

集群网络的配置作为 Cluster Network Operator (CNO) 配置的一部分被指定，并存储在名为 **cluster**的自定义资源（CR）对象中。CR 指定 **operator.openshift.io** API 组中的 **Network** API 的字段。

CNO 配置会在集群安装过程中从 **Network.config.openshift.io** API 组中的 **Network** API 继承以下字段，这些字段无法更改：

clusterNetwork

从中分配 pod IP 地址的 IP 地址池。

serviceNetwork

服务的 IP 地址池。

defaultNetwork.type

集群网络供应商，如 OpenShift SDN 或 OVN-Kubernetes。

您可以通过在名为 **cluster** 的 CNO 对象中设置 **defaultNetwork** 对象的字段来为集群指定集群网络供应商配置。

11.3.12.1. Cluster Network Operator 配置对象

Cluster Network Operator（CNO）的字段在下表中描述：

表 11.26. Cluster Network Operator 配置对象

字段	类型	描述
metadata.name	**字符串**	CNO 对象的名称。这个名称始终是 **cluster**。

spec.clusterNetwork	**数组**	用于指定从哪些 IP 地址块分配 Pod IP 地址以及分配给集群中每个节点的子网前缀长度的列表。例如：
		`spec:`
		`  clusterNetwork:`
		`  - cidr: 10.128.0.0/19`
		`  hostPrefix: 23`
		`  - cidr: 10.128.32.0/19`
		`  hostPrefix: 23`
		此值是只读的，并在 **install-config.yaml** 文件中指定。

spec.serviceNetwork	**数组**	服务的 IP 地址块。OpenShift SDN 和 OVN-Kubernetes Container Network Interface（CNI）网络供应商只支持服务网络具有单个 IP 地址块。例如：
		`spec:`
		`  serviceNetwork:`
		`  - 172.30.0.0/14`
		此值是只读的，并在 **install-config.yaml** 文件中指定。

spec.defaultNetwork	**对象**	为集群网络配置 Container Network Interface（CNI）集群网络供应商。
spec.kubeProxyConfig	**对象**	此对象的字段指定 kube-proxy 配置。如果您使用 OVN-Kubernetes 集群网络供应商，则 kube-proxy 的配置不会起作用。

defaultNetwork 对象配置
defaultNetwork 对象的值在下表中定义：

表 11.27. defaultNetwork 对象

字段	类型	描述

字段	类型	描述
type	**字符串**	**OpenShiftSDN** 或 **OVNKubernetes**。在安装过程中选择了集群网络供应商。集群安装后无法更改这个值。  **注意** OpenShift Container Platform 默认使用 OpenShift SDN Container Network Interface（CNI）集群网络供应商。
openshiftSDNConfig	**对象**	此对象仅对 OpenShift SDN 集群网络供应商有效。
ovnKubernetesConfig	**对象**	此对象仅对 OVN-Kubernetes 集群网络供应商有效。

配置 OpenShift SDN CNI 集群网络供应商
下表描述了 OpenShift SDN Container Network Interface（CNI）集群网络供应商的配置字段。

表 11.28. openshiftSDNConfig 对象

字段	类型	描述
mode	**字符串**	配置 OpenShift SDN 的网络隔离模式。默认值为 **NetworkPolicy**。 **Multitenant** 和 **Subnet** 的值可以向后兼容 OpenShift Container Platform 3.x，但不推荐这样做。集群安装后无法更改这个值。
mtu	**整数**	VXLAN 覆盖网络的最大传输单元 (MTU) 。这根据主网络接口的 MTU 自动探测。您通常不需要覆盖检测到的 MTU。 如果自动探测的值不是您期望的，请确认节点上主网络接口中的 MTU 是正确的。您不能使用这个选项更改节点上主网络接口的 MTU 值。 如果您的集群中的不同节点需要不同的 MTU 值，则必须将此值设置为比集群中的最低 MTU 值小 **50**。例如，如果集群中的某些节点的 MTU 为 **9001**，而某些节点的 MTU 为 **1500**，则必须将此值设置为 **1450**。 集群安装后无法更改这个值。

字段	类型	描述
vxlanPort	**整数**	用于所有 VXLAN 数据包的端口。默认值为 **4789**。集群安装后无法更改这个值。 如果您在虚拟环境中运行，并且现有节点是另一个 VXLAN 网络的一部分，那么可能需要更改此值。例如，当在 VMware NSX-T 上运行 OpenShift SDN 覆盖时，您必须为 VXLAN 选择一个备用端口，因为两个 SDN 都使用相同的默认 VXLAN 端口号。 在 Amazon Web Services (AWS) 上，您可以在端口 **9000** 和端口 **9999** 之间为 VXLAN 选择一个备用端口。

OpenShift SDN 配置示例

```
defaultNetwork:
  type: OpenShiftSDN
  openshiftSDNConfig:
    mode: NetworkPolicy
    mtu: 1450
    vxlanPort: 4789
```

配置 OVN-Kubernetes CNI 集群网络供应商

下表描述了 OVN-Kubernetes CNI 集群网络供应商的配置字段。

表 11.29. ovnKubernetesConfig 对象

字段	类型	描述
mtu	**整数**	Geneve（Generic Network Virtualization Encapsulation）覆盖网络的最大传输单元（MTU）。这根据主网络接口的 MTU 自动探测。您通常不需要覆盖检测到的 MTU。 如果自动探测的值不是您期望的，请确认节点上主网络接口中的 MTU 是正确的。您不能使用这个选项更改节点上主网络接口的 MTU 值。 如果您的集群中的不同节点需要不同的 MTU 值，则必须将此值设置为比集群中的最低 MTU 值小 **100**。例如，如果集群中的某些节点的 MTU 为 **9001**，而某些节点的 MTU 为 **1500**，则必须将此值设置为 **1400**。 集群安装后无法更改这个值。
genevePort	**整数**	用于所有 Geneve 数据包的端口。默认值为 **6081**。集群安装后无法更改这个值。

OVN-Kubernetes 配置示例

```
defaultNetwork:
  type: OVNKubernetes
```

```
ovnKubernetesConfig:
  mtu: 1400
  genevePort: 6081
```

kubeProxyConfig 对象配置
kubeProxyConfig 对象的值在下表中定义：

表 11.30. kubeProxyConfig 对象

字段	类型	描述
iptablesSyncPeriod	**字符串**	**iptables** 规则的刷新周期。默认值为 **30s**。有效的后缀包括 **s**、**m** 和 **h**，具体参见 Go **time** 软件包文档。 **注意** 由于 OpenShift Container Platform 4.3 及更高版本中引进了性能上的改进，现在不再需要调整 **iptablesSyncPeriod** 参数。

proxyArguments.iptables-min-sync-period	**数组**	刷新 **iptables** 规则前的最短时长。此字段确保刷新的频率不会过于频繁。有效的后缀包括 **s**、**m** 和 **h**，具体参见 Go **time** 软件包。默认值为：
		
		`kubeProxyConfig:`
		`  proxyArguments:`
		`  iptables-min-sync-period:`
		`  - 0s`

11.3.13. 部署集群

您可以在兼容云平台中安装 OpenShift Container Platform。

重要

安装程序的 **create cluster** 命令只能在初始安装过程中运行一次。

先决条件

- 获取 OpenShift Container Platform 安装程序以及集群的 pull secret。

流程

1. 更改为包含安装程序的目录并初始化集群部署：

```
$ ./openshift-install create cluster --dir <installation_directory> \ 1
    --log-level=info 2
```

1 对于 **<installation_directory>**，请指定自定义 **./install-config.yaml** 文件的位置。

2 要查看不同的安装详情，请指定 **warn**、**debug** 或 **error**，而不要指定 **info**。

集群部署完成后，终端会显示访问集群的信息，包括指向其 Web 控制台的链接和 **kubeadmin** 用户的凭证。

```
...
INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run 'export
KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'
INFO Access the OpenShift web-console here: https://console-openshift-
console.apps.mycluster.example.com
INFO Login to the console with user: "kubeadmin", and password: "4vYBz-Ee6gm-ymBZj-Wt5AL"
INFO Time elapsed: 36m22s
```

+

注意

当安装成功时，集群访问和凭证信息还会输出到 **<installation_directory>/.openshift_install.log**。

+

重要

- 安装程序生成的 Ignition 配置文件包含在 24 小时后过期的证书，然后在过期时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外情况是，您需要手动批准待处理的 **node-bootstrapper** 证书签名请求（CSR）来恢复 kubelet 证书。如需更多信息，请参阅*从过期的 control plane 证书中恢复*的文档。
- 建议您在生成 12 小时后使用 Ignition 配置文件，因为集群安装后 24 小时证书从 16 小时轮转至 22 小时。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中运行证书更新时避免安装失败。

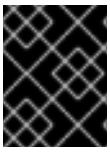
+

重要

您不得删除安装程序或安装程序所创建的文件。需要这两者才能删除集群。

11.3.14. 通过下载二进制文件安装 OpenShift CLI

您需要安装 CLI（**oc**）来使用命令行界面与 OpenShift Container Platform 进行交互。您可在 Linux、Windows 或 macOS 上安装 **oc**。



重要

如果安装了旧版本的 **oc**，则无法使用 OpenShift Container Platform 4.6 中的所有命令。下载并安装新版本的 **oc**。

11.3.14.1. 在 Linux 上安装 OpenShift CLI

您可以按照以下流程在 Linux 上安装 OpenShift CLI（**oc**）二进制文件。

流程

1. 进入到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 在 **Version** 下拉菜单中选择相应的版本。
3. 单击 **OpenShift v4.6 Linux 客户端** 条目旁边的 **Download Now**，再保存文件。
4. 解包存档：

```
$ tar xvzf <file>
```

5. 把 **oc** 二进制代码放到 **PATH** 中的目录中。
   执行以下命令可以查看当前的 **PATH** 设置：

```
$ echo $PATH
```

安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
$ oc <command>
```

11.3.14.2. 在 Windows 上安装 OpenShift CLI

您可以按照以下流程在 Windows 上安装 OpenShift CLI（**oc**）二进制代码。

流程

1. 进入到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 在 **Version** 下拉菜单中选择相应的版本。
3. 单击 **OpenShift v4.6 Windows 客户端**条目旁边的 **Download Now**，再保存文件。
4. 使用 ZIP 程序解压存档。
5. 把 **oc** 二进制代码放到 **PATH** 中的目录中。
   要查看您的 **PATH**，请打开命令提示窗口并执行以下命令：

```
C:\> path
```

安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
C:\> oc <command>
```

11.3.14.3. 在 macOS 上安装 OpenShift CLI

您可以按照以下流程在 macOS 上安装 OpenShift CLI（**oc**）二进制代码。

流程

1. 进入到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 在 **Version** 下拉菜单中选择相应的版本。
3. 单击 **OpenShift v4.6 MacOSX 客户端**条目旁边的 **Download Now**，再保存文件。
4. 解包和解压存档。
5. 将 **oc** 二进制文件移到 PATH 的目录中。
   要查看您的 **PATH**，打开一个终端窗口并执行以下命令：

```
$ echo $PATH
```

安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
$ oc <command>
```

11.3.15. 使用 CLI 登录到集群

您可以通过导出集群 **kubeconfig** 文件，以默认系统用户身份登录集群。**kubeconfig** 文件包含关于集群的信息，供 CLI 用于将客户端连接到正确集群和 API 服务器。该文件特只适用于一个特定的集群，在 OpenShift Container Platform 安装过程中创建。

先决条件

- 已部署了 OpenShift Container Platform 集群。
- 已安装 **oc** CLI。

流程

1. 导出 **kubeadmin** 凭证：

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

2. 使用导出的配置，验证能否成功运行 **oc** 命令：

```
$ oc whoami
```

输出示例

```
system:admin
```

11.3.16. 创建 registry 存储

安装集群后，必须为 registry Operator 创建存储。

11.3.16.1. 安装过程中删除的镜像 registry

在不提供可共享对象存储的平台上，OpenShift Image Registry Operator bootstraps 本身的状态是 **Removed**。这允许 **openshift-installer** 在这些平台类型上完成安装。

将 **ManagementState** Image Registry Operator 配置从 **Removed** 改为 **Managed**。

注意

Prometheus 控制台提供了一个 **ImageRegistryRemoved** 警报，例如：

"Image Registry has been removed.**ImageStreamTags**, **BuildConfigs** and **DeploymentConfigs** which reference **ImageStreamTags** may not work as expected.Please configure storage and update the config to **Managed** state by editing configs.imageregistry.operator.openshift.io."

11.3.16.2. 镜像 registry 存储配置

对于不提供默认存储的平台，Image Registry Operator 最初将不可用。安装后，您必须配置 registry 使用的存储，这样 Registry Operator 才可用。

示配置生产集群所需的持久性卷的说明。如果适用，显示有关将空目录配置为存储位置的说明，该位置只可用于非生产集群。

另外还提供了在升级过程中使用 **Recreate** rollout 策略来允许镜像 registry 使用块存储类型的说明。

11.3.16.2.1. 为 VMware vSphere 配置 registry 存储

作为集群管理员，在安装后需要配置 registry 来使用存储。

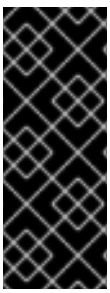
先决条件

- 具有 Cluster Administrator 权限
- VMware vSphere上有一个集群。
- 为集群置备的持久性存储，如 Red Hat OpenShift Container Storage。

重要

如果您只有一个副本，OpenShift Container Platform 支持对镜像 registry 存储的 **ReadWriteOnce** 访问。要部署支持高可用性的、带有两个或多个副本的镜像 registry，需要 **ReadWriteMany** 访问设置。

- 必须有“ 100Gi”容量。



重要

测试显示，在 RHEL 中使用 NFS 服务器作为核心服务的存储后端可能会出现问题。这包括 OpenShift Container Registry 和 Quay，Prometheus 用于监控存储，以及 Elasticsearch 用于日志存储。因此，不推荐使用 RHEL NFS 作为 PV 后端用于核心服务。

市场上的其他 NFS 实现可能没有这些问题。如需了解更多与此问题相关的信息，请联络相关的 NFS 厂商。

流程

1. 为了配置 registry 使用存储，需要修改 **configs.imageregistry/cluster** 资源中的 **spec.storage.pvc** 。

注意

使用共享存储时，请查看您的安全设置以防止被外部访问。

2. 验证您没有 registry pod:

```
$ oc get pod -n openshift-image-registry
```

注意

如果存储类型为 **emptyDIR**，则副本数不能超过 **1**。

3. 检查 registry 配置：

```
$ oc edit configs.imageregistry.operator.openshift.io
```

输出示例

```
storage:
  pvc:
    claim: 1
```

1 将 **claim** 字段留空以允许自动创建一个 **image-registry-storage** PVC。

4. 检查 **clusteroperator** 的状态：

```
$ oc get clusteroperator image-registry
```

11.3.16.2.2. 为 VMware vSphere 配置块 registry 存储

在作为集群管理员升级时，要允许镜像 registry 使用块存储类型，如 vSphere Virtual Machine Disk（VMDK），您可以使用 **Recreate** rollout 策略。

重要

支持块存储卷，但不建议将其用于生产环境中的镜像 registry。在块存储上配置 registry 的安装不具有高可用性，因为 registry 无法拥有多个副本。

流程

1. 要将镜像 registry 存储设置为块存储类型，对 registry 进行补丁，使其使用 **Recreate** rollout 策略，且仅使用 **1** 个副本运行：

```
$ oc patch config.imageregistry.operator.openshift.io/cluster --type=merge -p '{"spec":
{"rolloutStrategy":"Recreate","replicas":1}}'
```

2. 为块存储设备置备 PV，并为该卷创建 PVC。请求的块卷使用 ReadWriteOnce（RWO）访问模式。

   a. 创建包含以下内容的 **pvc.yaml** 文件以定义 VMware vSphere **PersistentVolumeClaim** :

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: image-registry-storage (1)
  namespace: openshift-image-registry (2)
spec:
  accessModes:
  - ReadWriteOnce (3)
  resources:
    requests:
      storage: 100Gi (4)
```

   (1) 代表 **PersistentVolumeClaim** 对象的唯一名称。

   (2) **PersistentVolumeClaim** 对象的命名空间，即 **openshift-image-registry**。

   (3) 持久性卷声明的访问模式。使用 **ReadWriteOnce** 时，单个节点可以通过读写权限挂载这个卷。

   (4) 持久性卷声明的大小。

   b. 从文件创建 **PersistentVolumeClaim** 对象：

```
$ oc create -f pvc.yaml -n openshift-image-registry
```

3. 编辑 registry 配置，使其可以正确引用 PVC:

```
$ oc edit config.imageregistry.operator.openshift.io -o yaml
```

输出示例

```
storage:
  pvc:
    claim: (1)
```

(1) 通过创建自定义 PVC，您可以将 **claim** 字段留空以用于默认自动创建 **image-registry-storage** PVC。

有关配置 registry 存储以便引用正确的 PVC 的说明，请参阅为 vSphere 配置 registry。

11.3.17. 备份 VMware vSphere 卷

OpenShift Container Platform 将新卷作为独立持久性磁盘置备，以便在集群中的任何节点上自由附加和分离卷。因此，无法备份使用快照的卷，也无法从快照中恢复卷。如需更多信息，请参阅 快照限制。

流程

要创建持久性卷的备份：

1. 停止使用持久性卷的应用程序。
2. 克隆持久性卷。
3. 重启应用程序。
4. 创建克隆的卷的备份。
5. 删除克隆的卷。

11.3.18. OpenShift Container Platform 的 Telemetry 访问

在 OpenShift Container Platform 4.6 中，默认运行的 Telemetry 服务提供有关集群健康状况和成功更新的指标，需要访问互联网。如果您的集群连接到互联网，Telemetry 会自动运行，而且集群会注册到 OpenShift Cluster Manager。

确认 OpenShift Cluster Manager 清单正确后，可以由 Telemetry 自动维护，也可以使用 OpenShift Cluster Manager 手动维护，使用订阅监控来跟踪帐户或多集群级别的 OpenShift Container Platform 订阅。

其他资源

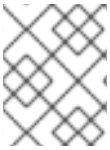
- 有关 Telemetry 服务的更多信息，请参阅关于远程健康监控。

11.3.19. 后续步骤

- 自定义集群。
- 如果需要，您可以选择不使用远程健康报告。
- 设置 registry 并配置 registry 存储。

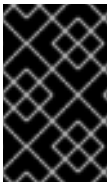
11.4. 使用用户置备的基础架构在 VSPHERE 上安装集群

在 OpenShift Container Platform 版本 4.6 中，您可以在您置备的 VMware vSphere 基础架构上安装集群。



注意

OpenShift Container Platform 支持将集群部署到单个 VMware vCenter 中。不支持在多个 vCenter 上使用机器/机器集部署集群。



重要

进行用户置备的基础架构安装的步骤仅作为示例。使用您提供的基础架构安装集群需要了解 vSphere 平台和 OpenShift Container Platform 的安装过程。使用用户置备的基础架构安装说明作为指南；您可以通过其他方法创建所需的资源。

11.4.1. 先决条件

- 为集群置备持久性存储。若要部署私有镜像 registry，您的存储必须提供 **ReadWriteMany** 访问模式。
- 查看有关 OpenShift Container Platform 安装和更新流程的详细信息。

- 完成安装要求您在 vSphere 主机上上传 Red Hat Enterprise Linux CoreOS(RHCOS)OVA。完成此过程的机器需要访问 vCenter 和 ESXi 主机上的端口 443。您确认可以访问端口 443。
- 如果您使用防火墙，您与管理员确认可以访问端口 443。control plane 节点必须能够通过端口 443 访问 vCenter 和 ESXi 主机，才能成功安装。
- 如果使用防火墙，则必须将其配置为允许集群需要访问的站点。

注意

如果您要配置代理，请务必也要查看此站点列表。

11.4.2. OpenShift Container Platform 的互联网访问

在 OpenShift Container Platform 4.6 中，您需要访问互联网来安装集群。

您必须具有以下互联网访问权限：

- 访问 OpenShift Cluster Manager 以下载安装程序并执行订阅管理。如果集群可以访问互联网，并且没有禁用 Telemetry，该服务会自动授权您的集群。
- 访问 Quay.io，以获取安装集群所需的软件包。
- 获取执行集群更新所需的软件包。

重要

如果您的集群无法直接访问互联网，则可以在置备的某些类基础架构上执行受限网络安装。在此过程中，您要下载所需的内容，并使用它在镜像 registry（mirror registry） 中填充安装集群并生成安装程序所需的软件包。对于某些安装类型，集群要安装到的环境不需要访问互联网。在更新集群之前，要更新 registry 镜像系统中的内容。

11.4.3. VMware vSphere 基础架构要求

您必须在满足您使用的组件要求的 VMware vSphere 版本 6 或 7 实例上安装 OpenShift Container Platform 集群。

表 11.31. VMware 组件支持的最低 vSphere 版本

组件	最低支持版本	描述
虚拟机监控程序	vSphere 6.5 及之后的版本 13	此版本是 Red Hat Enterprise Linux CoreOS(RHCOS)支持的最低版本。请查看 Red Hat Enterprise Linux 8 支持的管理程序列表。
使用 in-tree 驱动程序存储	vSphere 6.5 及之后的版本	此插件使用 OpenShift Container Platform 中包含的 vSphere 的树内存储驱动程序创建 vSphere 存储。

组件	最低支持版本	描述
可选： Networking (NSX-T)	vSphere 6.5U3 或 vSphere 6.7U2 及之后的版本	OpenShift Container Platform 需要 vSphere 6.5U3 或 vSphere 6.7U2+。VMware 的 NSX Container Plug-in（NCP）3.0.2 使用 OpenShift Container Platform 4.6 和 NSX-T 3.x+ 认证。

如果您使用 vSphere 版本 6.5 实例，请在安装 OpenShift Container Platform 前考虑升级到 6.7U3 或 7.0。

重要

您必须确保在安装 OpenShift Container Platform 前同步 ESXi 主机上的时间。请参阅 VMware 文档中的编辑主机时间配置。

11.4.4. 具有用户置备基础架构的集群的机器要求

对于含有用户置备的基础架构的集群，您必须部署所有所需的机器。

11.4.4.1. 所需的机器

最小的 OpenShift Container Platform 集群需要下列主机：

- 一个临时 bootstrap 机器
- 三台 control plane 或 master 机器
- 至少两台计算机器，也称为 worker 机器。

注意

集群要求 bootstrap 机器在三台 control plane 机器上部署 OpenShift Container Platform 集群。您可在安装集群后删除 bootstrap 机器。

重要

要保持集群的高可用性，请将独立的物理主机用于这些集群机器。

bootstrap 和 control plane 机器必须使用 Red Hat Enterprise Linux CoreOS (RHCOS) 作为操作系统。但是，计算机器可以在 Red Hat Enterprise Linux CoreOS(RHCOS)或 Red Hat Enterprise Linux(RHEL)7.9 之间进行选择。

请注意，RHCOS 基于 Red Hat Enterprise Linux（RHEL） 8，并继承其所有硬件认证和要求。请查看Red Hat Enterprise Linux 技术功能及限制 。

重要

所有虚拟机必须位于与安装程序相同的数据存储中。

11.4.4.2. 网络连接要求

所有 Red Hat Enterprise Linux CoreOS (RHCOS) 机器在启动过程中需要 **initramfs** 中的网络从 Machine Config Server 获取 Ignition 配置文件。在初次启动过程中，需要一个 DHCP 服务器或设置了静态 IP 地址来建立网络连接，以下载它们的 Ignition 配置文件。另外，集群中的每个 OpenShift Container Platform 节点都必须有权访问网络时间协议（NTP）服务器。如果 DHCP 服务器提供 NTP 服务器信息，Red Hat Enterprise Linux CoreOS（RHCOS）机器上的 chrony 时间服务会读取信息，并可与 NTP 服务器同步时钟。

11.4.4.3. 最低资源要求

每台集群机器都必须满足以下最低要求：

表 11.32. 最低资源要求

机器	操作系统	vCPU [1]	虚拟内存	存储	IOPS [2]
bootstrap	RHCOS	4	16 GB	100 GB	300
Control plane	RHCOS	4	16 GB	100 GB	300
Compute	RHCOS 或 RHEL 7.9	2	8 GB	100 GB	300

1. 当未启用并发多线程 (SMT) 或超线程时，一个 vCPU 相当于一个物理内核。启用后，使用以下公式来计算对应的比例：（每个内核数的线程）× sockets = vCPU。

2. OpenShift Container Platform 和 Kubernetes 对磁盘性能非常敏感，建议使用更快的存储速度，特别是 control plane 节点上需要 10 ms p99 fsync 持续时间的 etcd。请注意，在许多云平台上，存储大小和 IOPS 可一起扩展，因此您可能需要过度分配存储卷来获取足够的性能。

11.4.4.4. 证书签名请求管理

在使用您置备的基础架构时，集群只能有限地访问自动机器管理，因此您必须提供一种在安装后批准集群证书签名请求 (CSR) 的机制。**kube-controller-manager** 只能批准 kubelet 客户端 CSR。 **machine-approver** 无法保证使用 kubelet 凭证请求的提供证书的有效性，因为它不能确认是正确的机器发出了该请求。您必须决定并实施一种方法，以验证 kubelet 提供证书请求的有效性并进行批准。

11.4.5. 创建用户置备的基础架构

在部署采用用户置备的基础架构的 OpenShift Container Platform 集群前，您必须创建底层基础架构。

先决条件

- 在为集群创建支持基础架构之前，请参阅OpenShift Container Platform 4.x Tested Integrations页。

流程

1. 在每个节点上配置 DHCP 或设置静态 IP 地址。
2. 提供所需的负载均衡器。
3. 配置机器的端口。
4. 配置 DNS。
5. 确保网络可以正常工作。

11.4.5.1. 用户置备的基础架构对网络的要求

所有 Red Hat Enterprise Linux CoreOS（RHCOS）机器在启动过程中需要 **initramfs** 中的网络从机器配置服务器获取 Ignition 配置。

在初次启动过程中，需要一个 DHCP 服务器或集群中的每个机器都设置了静态 IP 地址来建立网络连接，以下载它们的 Ignition 配置文件。

建议您使用 DHCP 服务器为集群进行长期机器管理。确保 DHCP 服务器已配置为向集群机器提供持久 IP 地址和主机名。

Kubernetes API 服务器必须能够解析集群机器的节点名称。如果 API 服务器和 worker 节点位于不同的区域中，您可以配置默认 DNS 搜索区域，以便 API 服务器能够解析节点名称。另一种支持的方法是始终在节点对象和所有 DNS 请求中使用完全限定域名来指代主机。

您必须配置机器间的网络连接，以便集群组件进行通信。每台机器都必须能够解析集群中所有其他机器的主机名。

表 11.33. 所有机器到所有机器

协议	端口	描述
ICMP	N/A	网络可访问性测试
TCP	**1936**	指标
	9000-**9999**	主机级别的服务，包括端口 **9100**-**9101** 上的节点导出器和端口 **9099** 上的 Cluster Version Operator。
	10250-**10259**	Kubernetes 保留的默认端口
	10256	openshift-sdn
UDP	**4789**	VXLAN 和 Geneve
	6081	VXLAN 和 Geneve
	9000-**9999**	主机级别的服务，包括端口 **9100**-**9101** 上的节点导出器。
TCP/UDP	**30000**-**32767**	Kubernetes 节点端口

表 11.34. 要通过控制平面的所有机器

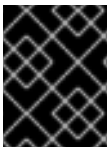
协议	端口	描述
TCP	**6443**	Kubernetes API

表 11.35. control plane 机器到 control plane 机器

协议	端口	描述
TCP	**2379-2380**	etcd 服务器和对等端口

网络拓扑要求

您为集群置备的基础架构必须满足下列网络拓扑要求。



重要

OpenShift Container Platform 要求所有节点都能访问互联网，以便为平台容器提取镜像并向红帽提供遥测数据。

负载均衡器

在安装 OpenShift Container Platform 前，您必须置备两个满足以下要求的负载均衡器：

1. **API 负载均衡器**：提供一个通用端点，供用户（包括人和机器）与平台交互和配置。配置以下条件：
   - 只适用于第 4 层负载均衡。这可被称为 Raw TCP、SSL Passthrough 或者 SSL 桥接模式。如果使用 SSL Bridge 模式，必须为 API 路由启用 Server Name Indication（SNI）。
   - 无状态负载平衡算法。这些选项根据负载均衡器的实现而有所不同。

重要

不要为 API 负载均衡器配置会话持久性。

在负载均衡器的前端和后台配置以下端口：

表 11.36. API 负载均衡器

端口	后端机器（池成员）	内部	外部	描述
6443	Bootstrap 和 control plane.bootstrap 机器初始化集群 control plane 后，您要从负载均衡器中删除 bootstrap 机器。您必须为 API 服务器健康检查探测配置 **/readyz** 端点。	X	X	Kubernetes API 服务器
22623	Bootstrap 和 control plane.bootstrap 机器初始化集群 control plane 后，您要从负载均衡器中删除 bootstrap 机器。	X		机器配置服务器

注意

负载均衡器必须配置为，从 API 服务器关闭 **/readyz** 端点到从池中删除 API 服务器实例时最多需要 30 秒。在 **/readyz** 返回错误或处于健康状态后的时间范围内，端点必须被删除或添加。每 5 秒或 10 秒探测一次，有两个成功请求处于健康状态，三个成为不健康的请求经过测试。

2. **应用程序入口负载均衡器**:提供来自集群外部的应用程序流量流量的 Ingress 点。配置以下条件：

   - 只适用于第 4 层负载均衡。这可被称为 Raw TCP、SSL Passthrough 或者 SSL 桥接模式。如果使用 SSL Bridge 模式，您必须为 Ingress 路由启用Server Name Indication（SNI）。
   - 建议根据可用选项以及平台上托管的应用程序类型，使用基于连接的或者基于会话的持久性。

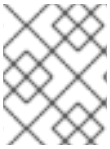
   在负载均衡器的前端和后台配置以下端口：

表 11.37. 应用程序入口负载均衡器

端口	后端机器（池成员）	内部	外部	描述
443	默认运行入口路由器 Pod、计算或 worker 的机器。	X	X	HTTPS 流量
80	默认运行入口路由器 Pod、计算或 worker 的机器。	X	X	HTTP 流量

提示

如果负载均衡器可以看到客户端的真实 IP 地址，启用基于 IP 的会话持久性可提高使用端到端 TLS 加密的应用程序的性能。



注意

OpenShift Container Platform 集群需要正确配置入口路由器。control plane 初始化后，您必须配置入口路由器。

以太网适配器硬件地址要求

当为集群置备虚拟机时，为每个虚拟机配置的以太网接口必须使用 VMware 机构唯一识别符 (OUI) 分配范围中的 MAC 地址：

- **00:05:69:00:00:00** 到 **00:05:69:FF:FF:FF**
- **00:0c:29:00:00:00** 到 **00:0c:29:FF:FF:FF**
- **00:1c:14:00:00:00** 到 **00:1c:14:FF:FF:FF**
- **00:50:56:00:00:00** 到 **00:50:56:FF:FF:FF**

如果使用 VMware OUI 以外的 MAC 地址，集群安装将无法成功。

NTP 配置

OpenShift Container Platform 集群默认配置为使用公共网络时间协议（NTP）服务器。如果要使用本地

OpenShift Container Platform 集群默认配置为使用公共网络时间协议（NTP）服务器。如果要使用本地企业 NTP 服务器，或者集群部署在断开连接的网络中，您可以将集群配置为使用特定的时间服务器。如需更多信息，请参阅*配置 chrony 时间服务*的文档。

如果 DHCP 服务器提供 NTP 服务器信息，Red Hat Enterprise Linux CoreOS（RHCOS）机器上的 chrony 时间服务会读取信息，并可与 NTP 服务器同步时钟。

其他资源

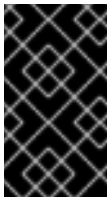
- 配置 chrony 时间服务

11.4.5.2. 用户置备 DNS 要求

DNS 用于名称解析和反向名称解析。DNS A/AAAA 或 CNAME 记录用于名称解析，PTR 记录用于反向解析名称。反向记录很重要，因为 Red Hat Enterprise Linux CoreOS（RHCOS）使用反向记录为所有节点设置主机名。另外，反向记录用于生成 OpenShift Container Platform 需要操作的证书签名请求（CSR）。

采用用户置备的基础架构的 OpenShift Container Platform 集群需要以下 DNS 记录。在每一记录中，**<cluster_name>** 是集群名称，**<base_domain>** 则是您在 **install-config.yaml** 文件中指定的集群基域。完整的 DNS 记录采用如下格式: **<component>.<cluster_name>.<base_domain>.**。

表 11.38. 所需的 DNS 记录

组件	记录	描述
Kubernetes API	**api.<cluster_name>.<base_domain>.**	添加 DNS A/AAAA 或 CNAME 记录，以及 DNS PTR 记录，以识别 control plane 机器的负载均衡器。这些记录必须由集群外的客户端以及集群中的所有节点解析。
	api-int.<cluster_name>.<base_domain>.	添加 DNS A/AAAA 或 CNAME 记录，以及 DNS PTR 记录，以识别 control plane 机器的负载均衡器。这些记录必须可以从集群中的所有节点解析。  **重要** API 服务器必须能够根据在 Kubernetes 中记录的主机名解析 worker 节点。如果 API 服务器无法解析节点名称，则代理的 API 调用会失败，且您无法从 pod 检索日志。
Routes	***.apps.<cluster_name>.<base_domain>.**	添加通配符 DNS A/AAAA 或 CNAME 记录，指向以运行入口路由器 Pod 的机器（默认为 worker 节点）为目标的负载均衡器。这些记录必须由集群外的客户端以及集群中的所有节点解析。
bootstrap	**bootstrap.<cluster_name>.<base_domain>.**	添加 DNS A/AAAA 或 CNAME 记录，以及 DNS PTR 记录来识别 bootstrap 机器。这些记录必须由集群中的节点解析。
Master 主机	**<master><n>.<cluster_name>.<base_domain>.**	DNS A/AAAA 或 CNAME 记录，以识别 control plane 节点（也称为 master 节点）的每台机器。这些记录必须由集群中的节点解析。

组件	记录	描述
Worker 主机	**<worker><n>. <cluster_name>. <base_domain>.**	添加 DNS A/AAAA 或 CNAME 记录，以识别 worker 节点的每台机器。这些记录必须由集群中的节点解析。

提示

您可以使用 **nslookup <hostname>** 命令来验证名称解析。您可以使用 **dig -x <ip_address>** 命令来验证 PTR 记录的反向名称解析。

下面的 BIND 区文件的例子展示了关于名字解析的 A 记录的例子。这个示例的目的是显示所需的记录。这个示例不是为选择一个名称解析服务提供建议。

例 11.7. DNS 区数据库示例

```
$TTL 1W
@ IN SOA ns1.example.com. root (
   2019070700 ; serial
   3H  ; refresh (3 hours)
   30M  ; retry (30 minutes)
   2W  ; expiry (2 weeks)
   1W )  ; minimum (1 week)
 IN NS ns1.example.com.
 IN MX 10 smtp.example.com.
;
;
ns1 IN A 192.168.1.5
smtp IN A 192.168.1.5
;
helper IN A 192.168.1.5
helper.ocp4 IN A 192.168.1.5
;
; The api identifies the IP of your load balancer.
api.ocp4  IN A 192.168.1.5
api-int.ocp4  IN A 192.168.1.5
;
; The wildcard also identifies the load balancer.
*.apps.ocp4  IN A 192.168.1.5
;
; Create an entry for the bootstrap host.
bootstrap.ocp4 IN A 192.168.1.96
;
; Create entries for the master hosts.
master0.ocp4  IN A 192.168.1.97
master1.ocp4  IN A 192.168.1.98
master2.ocp4  IN A 192.168.1.99
;
; Create entries for the worker hosts.
worker0.ocp4  IN A 192.168.1.11
```

```
worker1.ocp4  IN A 192.168.1.7
;
;EOF
```

下面的 BIND 区文件示例显示了反向名字解析的 PTR 记录示例。

例 11.8. 反向记录的 DNS 区数据库示例

```
$TTL 1W
@ IN SOA ns1.example.com. root (
   2019070700 ; serial
   3H  ; refresh (3 hours)
   30M  ; retry (30 minutes)
   2W  ; expiry (2 weeks)
   1W )  ; minimum (1 week)
 IN NS ns1.example.com.
;
; The syntax is "last octet" and the host must have an FQDN
; with a trailing dot.
97 IN PTR master0.ocp4.example.com.
98 IN PTR master1.ocp4.example.com.
99 IN PTR master2.ocp4.example.com.
;
96 IN PTR bootstrap.ocp4.example.com.
;
5 IN PTR api.ocp4.example.com.
5 IN PTR api-int.ocp4.example.com.
;
11 IN PTR worker0.ocp4.example.com.
7 IN PTR worker1.ocp4.example.com.
;
;EOF
```

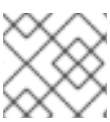
11.4.6. 生成 SSH 私钥并将其添加到代理中

如果要在集群上执行安装调试或灾难恢复，则必须为 **ssh-agent** 和安装程序提供 SSH 密钥。您可以使用此密钥访问公共集群中的 bootstrap 机器来排除安装问题。

注意

在生产环境中，您需要进行灾难恢复和调试。

您可以使用此密钥以 **core** 用户身份通过 SSH 连接到 master 节点。在部署集群时，此密钥会添加到 **core** 用户的 **~/.ssh/authorized_keys** 列表中。



注意

您必须使用一个本地密钥，而不要使用在特定平台上配置的密钥，如 AWS 密钥对。

流程

1. 如果还没有为计算机上免密码身份验证而配置的 SSH 密钥，请创建一个。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ ssh-keygen -t ed25519 -N '' \
  -f <path>/<file_name> 1
```

1 指定新 SSH 密钥的路径和文件名，如 **~/.ssh/id_rsa**。如果您已有密钥对，请确保您的公钥位于 **~/.ssh** 目录中。

运行此命令会在指定的位置生成不需要密码的 SSH 密钥。

注意

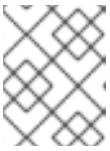
如果您计划在 **x86_64** 架构中安装使用 FIPS 验证的/Modules in Process 加密库的 OpenShift Container Platform 集群，不要创建使用 **ed25519** 算法的密钥。反之，创建一个使用 **rsa** 或 **ecdsa** 算法的密钥。

2. 作为后台任务启动 **ssh-agent** 进程：

```
$ eval "$(ssh-agent -s)"
```

输出示例

```
Agent pid 31874
```



注意

如果您的集群采用 FIPS 模式，则只使用 FIPS 兼容算法来生成 SSH 密钥。密钥必须是 RSA 或 ECDSA。

3. 将 SSH 私钥添加到 **ssh-agent**：

```
$ ssh-add <path>/<file_name> 1
```

输出示例

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

1 指定 SSH 私钥的路径和文件名，如 **~/.ssh/id_rsa**

后续步骤

- 在安装 OpenShift Container Platform 时，为安装程序提供 SSH 公钥。如果在您置备的基础架构上安装集群，您必须将此密钥提供给集群的机器。

11.4.7. 获取安装程序

在安装 OpenShift Container Platform 之前，将安装文件下载到本地计算机上。

先决条件

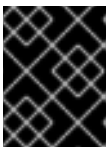
- 运行 Linux 或 macOS 的计算机，本地磁盘空间为 500 MB

流程

1. 访问 OpenShift Cluster Manager 站点的 Infrastructure Provider 页面。如果您有红帽帐号，请使用自己的凭证登录。如果没有，请创建一个帐户。
2. 选择您的基础架构供应商。
3. 进入适用于您的安装类型的页面，下载您的操作系统的安装程序，并将文件放在要保存安装配置文件的目录中。。

重要

安装程序会在用来安装集群的计算机上创建若干文件。在完成集群安装后，您必须保留安装程序和安装程序所创建的文件。这两个文件都需要删除集群。



重要

删除安装程序创建的文件不会删除您的集群，即使集群在安装过程中失败也是如此。要删除集群，为特定云供应商完成 OpenShift Container Platform 卸载流程。

4. 提取安装程序。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ tar xvf openshift-install-linux.tar.gz
```

5. 从 Red Hat OpenShift Cluster Manager 下载安装 pull secret 。通过此 pull secret，您可以进行所含授权机构提供的服务的身份验证，这些服务包括为 OpenShift Container Platform 组件提供容器镜像的 Quay.io。

11.4.8. 手动创建安装配置文件

对于使用用户置备的基础架构的 OpenShift Container Platform 安装，您必须手动生成安装配置文件。

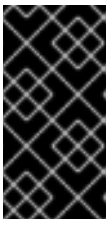
先决条件

- 获取 OpenShift Container Platform 安装程序和集群的访问令牌。

流程

1. 创建用来存储您所需的安装资产的安装目录：

```
$ mkdir <installation_directory>
```



重要

您必须创建目录。一些安装信息，如 bootstrap X.509 证书，有较短的过期间隔，因此不要重复使用安装目录。如果要重复使用另一个集群安装中的个别文件，可以将其复制到您的目录中。但是，一些安装数据的文件名可能会在发行版本之间有所改变。从 OpenShift Container Platform 老版本中复制安装文件时要格外小心。

2. 自定义以下 **install-config.yaml** 文件模板，并将它保存到 **<installation_directory>** 中。

注意

此配置文件必须命名为 **install-config.yaml**。

3. 备份 **install-config.yaml** 文件，以便用于安装多个集群。

重要

install-config.yaml 文件会在安装过程的下一步骤中消耗掉。现在必须备份它。

11.4.8.1. VMware vSphere install-config.yaml 文件示例

您可以自定义 **install-config.yaml** 文件，以指定有关 OpenShift Container Platform 集群平台的更多信息，或修改所需参数的值。

```
apiVersion: v1
baseDomain: example.com (1)
compute:
- hyperthreading: Enabled (2) (3)
  name: worker
  replicas: 0 (4)
controlPlane:
  hyperthreading: Enabled (5) (6)
  name: master
  replicas: 3 (7)
metadata:
  name: test (8)
platform:
  vsphere:
    vcenter: your.vcenter.server (9)
    username: username (10)
    password: password (11)
    datacenter: datacenter (12)
    defaultDatastore: datastore (13)
    folder: "/<datacenter_name>/vm/<folder_name>/<subfolder_name>" (14)
fips: false (15)
pullSecret: '{"auths": ...}' (16)
sshKey: 'ssh-ed25519 AAAA...' (17)
```

(1) 集群的基域。所有 DNS 记录都必须是这个基域的子域，并包含集群名称。

(2) (5) **controlPlane** 部分是一个单映射，但 compute 部分是一系列映射。为满足不同数据结构的要求，**compute** 部分的第一行必须以连字符 **-** 开头，**controlPlane** 部分的第一行则不可以连字符开头。虽然这两个部分目前都定义单个机器池，但未来的 OpenShift Container Platform 版本可能会支持在安装过程中定义多个计算池。只使用一个 control plane 池。

(3) (6) 是否要启用或禁用并发多线程或**超线程**。默认情况下，启用并发多线程以提高机器内核的性能。您可以通过将参数值设为 **Disabled** 来禁用。如果您在某些集群机器上禁用并发多线程，则必须在所有集群机器上禁用。

重要

如果禁用并发多线程，请确保在容量规划时考虑到机器性能可能会显著降低的问题。如果您禁用并发多线程，则计算机必须至少使用 8 个 CPU 和 32GB RAM。

4 **replicas** 参数的值必须设置为 **0**。此参数控制集群为您创建和管理的 worker 数量，使用用户置备的基础架构时集群不会执行这些功能。在完成 OpenShift Container Platform 安装前，您必须手动为集群部署 worker 机器。

7 您添加到集群的 control plane 机器数量。由于集群将这个值用作集群中 etcd 端点的数量，因此该值必须与您部署的 control plane 机器数量匹配。

8 您在 DNS 记录中指定的集群名称。

9 vCenter 服务器的完全限定主机名或 IP 地址。

10 用于访问服务器的用户名。此用户必须至少具有 vSphere 中 静态或动态持久性卷置备 所需的角色和权限。

11 与 vSphere 用户关联的密码。

12 vSphere 数据中心。

13 要使用的默认 vSphere 数据存储。

14 可选：对于安装程序置备的基础架构，安装程序创建虚拟机的现有文件夹的绝对路径，如 **/<datacenter_name>/vm/<folder_name>/<subfolder_name>**。如果没有提供这个值，安装程序会在数据中心虚拟机文件夹中创建一个顶层文件夹，其名称为基础架构 ID。如果您为集群提供基础架构,请省略此参数。

15 是否启用或禁用 FIPS 模式。默认情况下不启用 FIPS 模式。如果启用了 FIPS 模式，运行 OpenShift Container Platform 的 Red Hat Enterprise Linux CoreOS (RHCOS) 机器会绕过默认的 Kubernetes 加密套件，并使用由 RHCOS 提供的加密模块。

重要

只有在 **x86_64** 架构中的 OpenShift Container Platform 部署支持 FIPS 验证的/Modules in Process 加密库。

16 从 OpenShift Cluster Manager 获取的 pull secret。通过此 pull secret，您可以进行所含授权机构提供的服务的身份验证，这些服务包括为 OpenShift Container Platform 组件提供容器镜像的 Quay.io。

17 Red Hat Enterprise Linux CoreOS (RHCOS) 中 **core** 用户的默认 SSH 密钥的公钥部分。

11.4.8.2. 在安装过程中配置集群范围代理

生产环境可能会拒绝直接访问互联网，而是提供 HTTP 或 HTTPS 代理。您可以通过在 **install-config.yaml** 文件中配置代理设置，将新的 OpenShift Container Platform 集群配置为使用代理。

先决条件

- 您有一个现有的 **install-config.yaml** 文件。

- 您检查了集群需要访问的站点，并决定是否需要绕过代理。默认情况下代理所有集群出口流量，包括对托管云供应商 API 的调用。您需要将站点添加到 **Proxy** 对象的 **spec.noProxy** 字段来绕过代理。

注意

Proxy 对象 **status.noProxy** 字段使用安装配置中的 **networking.machineNetwork[].cidr**、**networking.clusterNetwork[].cidr** 和 **networking.serviceNetwork[]** 字段的值填充。

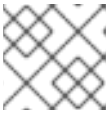
对于在 Amazon Web Services(AWS)、Google Cloud Platform(GCP)、Microsoft Azure 和 Red Hat OpenStack Platform(RHOSP)上安装, **Proxy** 对象 **status.noProxy** 字段也会使用实例元数据端点填充(**169.254.169.254**)。

流程

1. 编辑 **install-config.yaml** 文件并添加代理设置。例如：

```
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: example.com 3
additionalTrustBundle: | 4
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
...
```

1. 用于创建集群外 HTTP 连接的代理 URL。URL 必须是 **http**。
2. 用于创建集群外 HTTPS 连接的代理 URL。
3. 要排除在代理中的目标域名、IP 地址或其他网络 CIDR 的逗号分隔列表。在域前面加 **.** 来仅匹配子域。例如： **.y.com** 匹配 **x.y.com**，但不匹配 **y.com**。使用 * 绕过所有目的地的代理。您必须包含 vCenter 的 IP 地址以及用于其机器的 IP 范围。
4. 如果提供，安装程序会在 **openshift-config 命名空间中生成名为 user-ca- bundle** 的配置映射来保存额外的 CA 证书。如果您提供 **additionalTrustBundle** 和至少一个代理设置，则 **Proxy** 对象会被配置为引用 **trustedCA** 字段中的 **user-ca-bundle** 配置映射。然后，Cluster Network Operator 会创建一个 **trusted-ca-bundle** 配置映射，该配置映射将为 **trustedCA** 参数指定的内容与 RHCOS 信任捆绑包合并。 **additionalTrustBundle** 字段是必需的，除非代理的身份证书由来自 RHCOS 信任捆绑包的颁发机构签名。

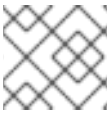


注意

安装程序不支持代理的 **readinessEndpoints** 字段。

2. 保存该文件，并在安装 OpenShift Container Platform 时引用。

安装程序会创建一个名为 **cluster** 的集群范围代理，该代理使用提供的 **install-config.yaml** 文件中的代理设置。如果没有提供代理设置，仍然会创建一个 **cluster Proxy** 对象，但它会有一个空 **spec**。



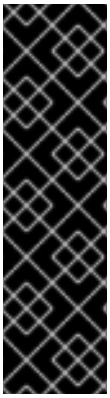
注意

只支持名为 **cluster** 的 **Proxy** 对象，且无法创建额外的代理。

11.4.9. 创建 Kubernetes 清单和 Ignition 配置文件

由于您必须修改一些集群定义文件并要手动启动集群机器，因此您必须生成 Kubernetes 清单和 Ignition 配置文件，集群需要这两项来创建其机器。

安装配置文件转换为 Kubernetes 清单。清单嵌套到 Ignition 配置文件中，稍后用于创建集群。



重要

- 安装程序生成的 Ignition 配置文件包含在 24 小时后过期的证书，然后在过期时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外情况是，您需要手动批准待处理的 **node-bootstrapper** 证书签名请求（CSR）来恢复 kubelet 证书。如需更多信息，请参阅*从过期的 control plane 证书中恢复*的文档。
- 建议您在生成 12 小时后使用 Ignition 配置文件，因为集群安装后 24 小时证书从 16 小时轮转至 22 小时。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中运行证书更新时避免安装失败。

先决条件

- 已获得 OpenShift Container Platform 安装程序。
- 已创建 **install-config.yaml** 安装配置文件。

流程

1. 切换到包含安装程序的目录，并为集群生成 Kubernetes 清单：

```
$ ./openshift-install create manifests --dir <installation_directory> 1
```

1 对于 **<installation_directory>**，请指定含有您创建的 **install-config.yaml** 文件的安装目录。

2. 删除定义 control plane 机器的 Kubernetes 清单文件以及计算机器集：

```
$ rm -f openshift/99_openshift-cluster-api_master-machines-*.yaml openshift/99_openshift-cluster-api_worker-machineset-*.yaml
```

由于您要自行创建和管理这些资源，因此不必初始化这些资源。

- 您可以使用机器 API 来保留机器集文件来创建计算机器，但您必须更新对其的引用，以匹配您的环境。

3. 检查 **<installation_directory>/manifests/cluster-scheduler-02-config.yml** Kubernetes 清单文件中的 **mastersSchedulable** 参数是否已设置为 **false**。此设置可防止在 control plane 机器上调度 pod:

a. 打开 **<installation_directory>/manifests/cluster-scheduler-02-config.yml** 文件。

b. 找到 **mastersSchedulable** 参数并确保它被设置为 **false**。

c. 保存并退出文件。

4. 要创建 Ignition 配置文件，从包含安装程序的目录运行以下命令：

```
$ ./openshift-install create ignition-configs --dir <installation_directory> 1
```

1 对于 **<installation_directory>**，请指定相同的安装目录。

该目录中将生成以下文件：

```
.
├── auth
│   ├── kubeadmin-password
│   └── kubeconfig
├── bootstrap.ign
├── master.ign
├── metadata.json
└── worker.ign
```

11.4.10. 提取基础架构名称

Ignition 配置文件包含一个唯一的集群标识符,您可以使用它在 VMware vSphere 中唯一地标识您的集群。如果计划使用集群标识符作为虚拟机文件夹的名称,您必须提取它。

先决条件

- 获取 OpenShift Container Platform 安装程序以及集群的 pull secret。
- 已为集群生成 Ignition 配置文件。
- 安装了 **jq** 软件包。

流程

- 要从 Ignition 配置文件元数据中提取和查看基础架构名称，请运行以下命令：

```
$ jq -r .infraID <installation_directory>/metadata.json 1
```

1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

输出示例

```
openshift-vw9j6 1
```

1 此命令的输出是您的集群名称和随机字符串。

11.4.11. 在 vSphere 中创建 Red Hat Enterprise Linux CoreOS (RHCOS) 机器

在 VMware vSphere 上安装包含用户置备基础架构的集群前，您必须在 vSphere 主机上创建 RHCOS 机器供其使用。

先决条件

- 已获取集群的 Ignition 配置文件。
- 您可以从计算机访问 HTTP 服务器，并且您创建的机器也可以访问该服务器。
- 您已创建了 vSphere 集群。

流程

1. 将名为 **<installation_directory>/bootstrap.ign** 的 bootstrap Ignition 配置文件上传到 HTTP 服务器，该配置文件是由安装程序创建的。记下此文件的 URL。
2. 将 bootstrap 节点的以下辅助 Ignition 配置文件保存到计算机中，存为 **<installation_directory>/merge-bootstrap.ign** ：

```
{
  "ignition": {
    "config": {
      "merge": [
        {
          "source": "<bootstrap_ignition_config_url>", 1
          "verification": {}
        }
      ]
    },
    "timeouts": {},
    "version": "3.1.0"
  },
  "networkd": {},
  "passwd": {},
  "storage": {},
  "systemd": {}
}
```

1 指定您托管的 bootstrap Ignition 配置文件的 URL。

为 bootstrap 机器创建虚拟机 (VM) 时，您要使用此 Ignition 配置文件。

3. 找到安装程序创建的以下 Ignition 配置文件：
   - **<installation_directory>/master.ign**
   - **<installation_directory>/worker.ign**
   - **<installation_directory>/merge-bootstrap.ign**
4. 将 Ignition 配置文件转换为 Base64 编码。在此流程中，您必须将这些文件添加到虚拟机中的额外配置参数 **guestinfo.ignition.config.data** 中。
   例如，如果您使用 Linux 操作系统，可以使用 **base64** 命令来编码这些文件。

```
$ base64 -w0 <installation_directory>/master.ign > <installation_directory>/master.64
```

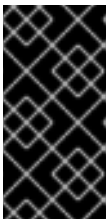
```
$ base64 -w0 <installation_directory>/worker.ign > <installation_directory>/worker.64
```

```
$ base64 -w0 <installation_directory>/merge-bootstrap.ign > <installation_directory>/merge-bootstrap.64
```

重要

如果您计划在安装完成后在集群中添加更多计算机器，请不要删除这些文件。

5. 获取 RHCOS OVA 镜像。镜像位于 RHCOS 镜像镜像页面。

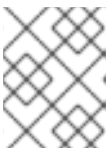


重要

RHCOS 镜像可能不会随着 OpenShift Container Platform 的每一发行版本都有改变。您必须下载一个最高版本的镜像，其版本号应小于或等于您安装的 OpenShift Container Platform 版本。如果可用，请使用与 OpenShift Container Platform 版本匹配的镜像版本。

文件名包含 OpenShift Container Platform 版本号，格式为 **rhcos-vmware.<architecture>.ova**。

6. 在 vSphere 客户端中，在数据中心中创建一个文件夹来存储您的虚拟机。
   a. 点击 **VMs and Templates** 视图。
   b. 右键点击您的数据中心名称。
   c. 点击 **New Folder → New VM and Template Folder**。
   d. 在显示的窗口中输入文件夹名称。如果您没有在 **install-config.yaml** 文件中指定现有文件夹，请创建一个文件夹，其名称与基础架构 ID 相同。您可以使用这个文件夹名称，因此 vCenter 会在适当的位置为 Workspace 配置动态置备存储。
7. 在 vSphere 客户端中，为 OVA 镜像创建一个模板，然后根据需要克隆模板。



注意

在以下步骤中，您将创建一个模板，然后克隆所有集群机器的模板。然后，在置备虚拟机时，为该克隆的机器类型提供 Ignition 配置文件的位置。

   a. 在 **Hosts and Clusters** 选项卡中，右键点击您的集群名称并选择 **Deploy OVF Template**。
   b. 在 **Select an OVF** 选项卡中，指定您下载的 RHCOS OVA 文件的名称。
   c. 在 **Select a name and folder** 选项卡中，为您的模板设置 **虚拟机名称**，如 **Template-RHCOS**。点击 vSphere 集群的名称并选择您在上一步中创建的文件夹。
   d. 在 **Select a compute resource** 选项卡中，点击您的 vSphere 集群名称。
   e. 在 **Select storage** 选项卡中，配置虚拟机的存储选项。
      - 根据您的存储要求，选择**Thin Provision** 或 **Thick Provision**。
      - 选择您在 **install-config.yaml** 文件中指定的数据存储。

f. 在 **Select network** 选项卡中，指定您为集群配置的网络（如果可用）。

g. 在创建 OVF 模板时，请不要在 **Customize template** 选项卡上指定值，或者不要再配置模板。

重要

不要启动原始虚拟机模板。VM 模板必须保持关闭状态，必须为新的 RHCOS 机器克隆。启动虚拟机模板会将虚拟机模板配置为平台上的虚拟机，这样可防止它被用作计算机集可以应用配置的模板。

8. 部署模板后，为集群中的机器部署虚拟机。

a. 右键点击模板的名称，再点击 **Clone → Clone to Virtual Machine**。

b. 在 **Select a name and folder**选项卡中，指定虚拟机的名称。名称中可以包括机器类型，如 **control-plane-0** 或 **compute-1**。

c. 在 **Select a name and folder**选项卡中，选择您为集群创建的文件夹名称。

d. 在 **Select a compute resource** 选项卡中，选择数据中心中的主机名称。
对于 bootstrap 机器，指定您托管的 bootstrap Ignition 配置文件的 URL。

e. 可选：在 **Select storage** 选项卡中，自定义存储选项。

f. 在 **Select clone options** 中，选择 **Customize this virtual machine's hardware**。

g. 在 **Customize hardware** 选项卡中，点击 **VM Options → Advanced**。

- 可选：覆盖 vSphere 中的默认 DHCP 网络。启用静态 IP 网络：

i. 设置静态 IP 配置：

```
$ export IPCFG="ip=<ip>::<gateway>:<netmask>:<hostname>:<iface>:none
nameserver=srv1 [nameserver=srv2 [nameserver=srv3 [...]]]"
```

示例命令

```
$ export IPCFG="ip=192.168.100.101::192.168.100.254:255.255.255.0:::none
nameserver=8.8.8.8"
```

ii. 在从 vSphere 中的 OVA 引导虚拟机前，设置 **guestinfo.afterburn.initrd.network-kargs** 属性：

```
$ govc vm.change -vm "<vm_name>" -e "guestinfo.afterburn.initrd.network-
kargs=${IPCFG}"
```

- 可选：在出现集群性能问题时，从 **Latency Sensitivity** 列表中选择 **High**。确定虚拟机的 CPU 和内存保留有以下值：

  - 内存保留值必须等于其配置的内存大小。
  - CPU 保留值必须至少是低延迟虚拟 CPU 的数量，乘以测量的物理 CPU 速度。

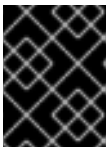
点击 **Edit Configuration**，然后在 **Configuration Parameters** 窗口中点击 **Add**

- 点击 **Edit Configuration**，然后在 **Configuration Parameters** 窗口中点击 **Add Configuration Params**。定义以下参数名称和值：
  - **guestinfo.ignition.config.data** ：找到您在此流程中创建的 base-64 编码文件，并粘贴此机器类型的 base64 编码 Ignition 配置文件的内容。
  - **guestinfo.ignition.config.data.encoding**：指定 **base64**。
  - **disk.EnableUUID**：指定 **TRUE**。

h. 在 **Customize hardware** 选项卡的 **Virtual Hardware** 面板中，根据需要修改指定的值。确保 RAM、CPU 和磁盘存储的数量满足机器类型的最低要求。

i. 完成配置并打开虚拟机电源。

9. 对于每台机器，按照前面的步骤为集群创建其余的机器。



重要

此刻您必须创建 bootstrap 和 control plane 机器。由于计算机器中已默认部署了一些 Pod，因此在安装集群前，还要创建至少两台计算机器。

11.4.12. 在 vSphere 中创建更多 Red Hat Enterprise Linux CoreOS（RHCOS）机器

您可以为集群创建更多计算机器，在 VMware vSphere 上使用用户置备的基础架构。

先决条件

- 获取计算机器的 Base64 编码 Ignition 文件。
- 您可以访问您为集群创建的 vSphere 模板。

流程

1. 部署模板后，为集群中的机器部署虚拟机。

a. 右键点击模板的名称，再点击 **Clone → Clone to Virtual Machine**。

b. 在 **Select a name and folder**选项卡中，指定虚拟机的名称。您可以在名称中包含机器类型，如 **compute-1**。

c. 在 **Select a name and folder**选项卡中，选择您为集群创建的文件夹名称。

d. 在 **Select a compute resource** 选项卡中，选择数据中心中的主机名称。

e. 可选：在 **Select storage** 选项卡中，自定义存储选项。

f. 在 **Select clone options** 中，选择 **Customize this virtual machine's hardware**。

g. 在 **Customize hardware** 选项卡中，点击 **VM Options → Advanced**。

- 从 **Latency Sensitivity** 列表中选择 **High**。
- 点击 **Edit Configuration**，然后在 **Configuration Parameters** 窗口中点击 **Add Configuration Params**。定义以下参数名称和值：

- **guestinfo.ignition.config.data**：粘贴此机器类型的 Base64 编码计算 Ignition 配置文件的内容。
- **guestinfo.ignition.config.data.encoding**：指定 **base64**。
- **disk.EnableUUID**：指定 **TRUE**。

h. 在 **Customize hardware** 选项卡的 **Virtual Hardware** 面板中，根据需要修改指定的值。确保 RAM、CPU 和磁盘存储的数量满足机器类型的最低要求。另外，如果有多个可用的网络，请确定在 **Add network adapter** 中选择正确的网络。

i. 完成配置并打开虚拟机电源。

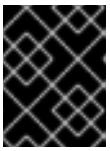
2. 继续为集群创建更多计算机器。

11.4.13. 磁盘分区

在大多数情况下，数据分区最初是由安装 RHCOS 而不是安装另一个操作系统来创建的。在这种情况下，OpenShift Container Platform 安装程序应该被允许配置磁盘分区。

但是，在安装 OpenShift Container Platform 节点时，在两种情况下您可能需要覆盖默认分区：

- 创建单独的分区：对于在空磁盘中的 greenfield 安装，您可能想要在分区中添加单独的存储。这只在生成 **/var** 或者一个 **/var** 独立分区的子目录（如 **/var/lib/etcd** ）时被正式支持，但不支持两者。



重要

Kubernetes 只支持两个文件系统分区。如果您在原始配置中添加多个分区，Kubernetes 无法监控所有这些分区。

- 保留现有分区：对于 brownfield 安装，您要在现有节点上重新安装 OpenShift Container Platform，并希望保留从之前的操作系统中安装的数据分区，对于 **coreos-installer** 来说，引导选项和选项都允许您保留现有数据分区。

创建一个独立的 /var 分区

通常情况下，OpenShift Container Platform 的磁盘分区应该留给安装程序。然而，在有些情况下您可能需要在文件系统的一部分中创建独立分区。

OpenShift Container Platform 支持添加单个分区来将存储附加到 **/var** 分区或 **/var** 的子目录。例如：

- **/var/lib/containers**：保存镜像相关的内容，随着更多镜像和容器添加到系统中，它所占用的存储会增加。
- **/var/lib/etcd**：保存您可能希望保持独立的数据，比如 etcd 存储的性能优化。
- **/var**：保存您希望独立保留的数据，用于特定目的（如审计）。

单独存储 **/var** 目录的内容可方便地根据需要对区域扩展存储，并可以在以后重新安装 OpenShift Container Platform 时保持该数据地完整。使用这个方法，您不必再次拉取所有容器，在更新系统时也无法复制大量日志文件。

因为 **/var** 在进行一个全新的 Red Hat Enterprise Linux CoreOS（RHCOS）安装前必需存在，所以这个流程会在 OpenShift Container Platform 安装过程的 **openshift-install** 准备阶段插入的机器配置来设置独立的 **/var** 分区。

流程

1. 创建存放 OpenShift Container Platform 安装文件的目录：

```
$ mkdir $HOME/clusterconfig
```

2. 运行 **openshift-install** 在 **manifest** 和 **openshift** 子目录中创建一组文件。在出现提示时回答系统问题：

```
$ openshift-install create manifests --dir $HOME/clusterconfig
? SSH Public Key ...
$ ls $HOME/clusterconfig/openshift/
99_kubeadmin-password-secret.yaml
99_openshift-cluster-api_master-machines-0.yaml
99_openshift-cluster-api_master-machines-1.yaml
99_openshift-cluster-api_master-machines-2.yaml
...
```

3. 创建 **MachineConfig** 对象并将其添加到 **openshift** 目录中的一个文件中。例如，把文件命名为 **98-var-partition.yaml**，将磁盘设备名称改为 **worker** 系统中存储设备的名称，并根据情况设置存储大小。这个示例将 **/var** 目录放在一个单独的分区中：

```
apiVersion: machineconfiguration.openshift.io/v1
kind: MachineConfig
metadata:
  labels:
    machineconfiguration.openshift.io/role: worker
  name: 98-var-partition
spec:
  config:
    ignition:
      version: 3.1.0
    storage:
      disks:
      - device: /dev/<device_name> 1
        partitions:
        - label: var
          startMiB: <partition_start_offset> 2
          sizeMiB: <partition_size> 3
      filesystems:
        - device: /dev/disk/by-partlabel/var
          path: /var
          format: xfs
    systemd:
      units:
        - name: var.mount 4
          enabled: true
          contents: |
            [Unit]
            Before=local-fs.target
            [Mount]
            What=/dev/disk/by-partlabel/var
            Where=/var
```

```
Options=defaults,prjquota 5
[Install]
WantedBy=local-fs.target
```

1. 要分区的磁盘的存储设备名称。
2. 当在引导磁盘中添加数据分区时，推荐最少使用 25000MB。root 文件系统会自动重新定义大小使其占据所有可用空间（最多到指定的偏移值）。如果没有指定值，或者指定的值小于推荐的最小值，则生成的 root 文件系统会太小，而在以后进行的 RHCOS 重新安装可能会覆盖数据分区的开始部分。
3. 数据分区的大小（以兆字节为单位）。
4. 挂载单元的名称必须与 **Where=** 指令中指定的目录匹配。例如，对于挂载在 **/var/lib/containers** 上的文件系统，该单元必须命名为 **var-lib-containers.mount**。
5. 对于用于容器存储的文件系统，必须启用 **prjquota** 挂载选项。

注意

在创建独立 **/var** 分区时，如果不同的实例类型没有相同的设备名称，则无法将不同的实例类型用于 worker 节点。

4. 再次运行 **openshift-install**，从 **manifest** 和 **openshift** 子目录中的一组文件创建 Ignition 配置：

```
$ openshift-install create ignition-configs --dir $HOME/clusterconfig
$ ls $HOME/clusterconfig/
auth  bootstrap.ign  master.ign  metadata.json  worker.ign
```

现在，您可以使用 Ignition 配置文件作为 vSphere 安装程序的输入来安装 Red Hat Enterprise Linux CoreOS（RHCOS）系统。

11.4.14. 通过下载二进制文件安装 OpenShift CLI

您需要安装 CLI（**oc**） 来使用命令行界面与 OpenShift Container Platform 进行交互。您可在 Linux、Windows 或 macOS 上安装 **oc**。

重要

如果安装了旧版本的 **oc**，则无法使用 OpenShift Container Platform 4.6 中的所有命令。下载并安装新版本的 **oc**。

11.4.14.1. 在 Linux 上安装 OpenShift CLI

您可以按照以下流程在 Linux 上安装 OpenShift CLI（**oc**）二进制文件。

流程

1. 进入到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 在 **Version** 下拉菜单中选择相应的版本。
3. 单击 **OpenShift v4.6 Linux 客户端** 条目旁边的 **Download Now**，再保存文件。

4. 解包存档：

```
$ tar xvzf <file>
```

5. 把 **oc** 二进制代码放到 **PATH** 中的目录中。
执行以下命令可以查看当前的 **PATH** 设置：

```
$ echo $PATH
```

安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
$ oc <command>
```

11.4.14.2. 在 Windows 上安装 OpenShift CLI

您可以按照以下流程在 Windows 上安装 OpenShift CLI（**oc**）二进制代码。

流程

1. 进入到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 在 **Version** 下拉菜单中选择相应的版本。
3. 单击 **OpenShift v4.6 Windows 客户端**条目旁边的 **Download Now**，再保存文件。
4. 使用 ZIP 程序解压存档。
5. 把 **oc** 二进制代码放到 **PATH** 中的目录中。
要查看您的 **PATH**，请打开命令提示窗口并执行以下命令：

```
C:\> path
```

安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
C:\> oc <command>
```

11.4.14.3. 在 macOS 上安装 OpenShift CLI

您可以按照以下流程在 macOS 上安装 OpenShift CLI（**oc**）二进制代码。

流程

1. 进入到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 在 **Version** 下拉菜单中选择相应的版本。
3. 单击 **OpenShift v4.6 MacOSX 客户端**条目旁边的 **Download Now**，再保存文件。
4. 解包和解压存档。
5. 将 **oc** 二进制文件移到 PATH 的目录中。
要查看您的 **PATH**，打开一个终端窗口并执行以下命令：

```
$ echo $PATH
```

安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
$ oc <command>
```

11.4.15. 创建集群

要创建 OpenShift Container Platform 集群，请等待您通过安装程序生成的 Ignition 配置文件所置备的机器上完成 bootstrap 过程。

先决条件

- 为集群创建所需的基础架构。
- 已获得安装程序并为集群生成了 Ignition 配置文件。
- 已使用 Ignition 配置文件为集群创建 RHCOS 机器。
- 您的机器可直接访问互联网，或者可以使用 HTTP 或 HTTPS 代理。

流程

1. 监控 bootstrap 过程：

```
$ ./openshift-install --dir <installation_directory> wait-for bootstrap-complete \ 1
    --log-level=info 2
```

1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

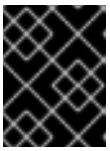
2 要查看不同的安装详情，请指定 **warn**、**debug** 或 **error**，而不要指定 **info**。

输出示例

```
INFO Waiting up to 30m0s for the Kubernetes API at https://api.test.example.com:6443...
INFO API v1.19.0 up
INFO Waiting up to 30m0s for bootstrapping to complete...
INFO It is now safe to remove the bootstrap resources
```

Kubernetes API 服务器提示已在 control plane 机器上完成 bootstrap 时，命令运行成功。

2. bootstrap 过程完成后，请从负载均衡器中删除 bootstrap 机器。



重要

此时您必须从负载均衡器中删除 bootstrap 机器。您还可以删除或重新格式化机器本身。

11.4.16. 使用 CLI 登录到集群

您可以通过导出集群 **kubeconfig** 文件，以默认系统用户身份登录集群。**kubeconfig** 文件包含关于集群的信息，供 CLI 用于将客户端连接到正确集群和 API 服务器。该文件特只适用于一个特定的集群，在 OpenShift Container Platform 安装过程中创建。

先决条件

- 已部署了 OpenShift Container Platform 集群。
- 已安装 **oc** CLI。

流程

1. 导出 **kubeadmin** 凭证：

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

2. 使用导出的配置，验证能否成功运行 **oc** 命令：

```
$ oc whoami
```

输出示例

```
system:admin
```

11.4.17. 批准机器的证书签名请求

将机器添加到集群时，会为您添加的每台机器生成两个待处理证书签名请求（CSR）。您必须确认这些 CSR 已获得批准，或根据需要自行批准。客户端请求必须首先被批准，然后是服务器请求。

先决条件

- 您已将机器添加到集群中。

流程

1. 确认集群可以识别这些机器：

```
$ oc get nodes
```

输出示例

```
NAME     STATUS   ROLES   AGE  VERSION
master-0 Ready    master  63m  v1.19.0
master-1 Ready    master  63m  v1.19.0
master-2 Ready    master  64m  v1.19.0
```

输出将列出您创建的所有机器。

注意

在一些 CSR 被批准前，以上输出可能不包括计算节点（也称为 worker 节点）。

2. 检查待处理的 CSR，并确保可以看到添加到集群中的每台机器都有 **Pending** 或 **Approved** 状态的客户端请求：

```
$ oc get csr
```

输出示例

```
NAME        AGE    REQUESTOR                                                   CONDITION
csr-8b2br   15m    system:serviceaccount:openshift-machine-config-operator:node-bootstrapper   Pending
csr-8vnps   15m    system:serviceaccount:openshift-machine-config-operator:node-bootstrapper   Pending
...
```

在本例中，两台机器加入了集群。您可能会在列表中看到更多已批准的 CSR。

3. 如果 CSR 没有获得批准，请在所添加机器的所有待处理 CSR 都处于 **Pending** 状态后，为您的集群机器批准这些 CSR：

注意

由于 CSR 会自动轮转，因此请在将机器添加到集群后一小时内批准您的 CSR。如果没有在一小时内批准，证书将会轮转，每个节点将会存在多个证书。您必须批准所有这些证书。批准客户端 CSR 后， Kubelet 为服务证书创建辅助 CSR，这需要手动批准。然后，如果 Kubelet 请求具有相同参数的新证书，则 **machine-approver** 会自动批准后续服务证书续订请求。

注意

对于在未启用机器 API 的平台中运行的集群，如裸机和其他用户置备的基础架构，必须采用一种方法自动批准 kubelet 提供证书请求（CSR）。如果没有批准请求，则 **oc exec**、**oc rsh** 和 **oc logs** 命令将无法成功，因为 API 服务器连接到 kubelet 时需要服务证书。与 Kubelet 端点联系的任何操作都需要此证书批准。这个方法必须监视新的 CSR，确认 CSR 由 **system:node** 或 **system:admin** 组中的 **node-bootstrapper** 服务帐户提交，并确认节点的身份。

- 若要单独批准，请对每个有效的 CSR 运行以下命令：

```
$ oc adm certificate approve <csr_name> 1
```

1 **<csr_name>** 是当前 CSR 列表中 CSR 的名称。

- 要批准所有待处理的 CSR，请运行以下命令：

```
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}{{"\n"}}{{end}}{{end}}' | xargs --no-run-if-empty oc adm certificate approve
```

注意

在有些 CSR 被批准前，一些 Operator 可能无法使用。

4. 现在，您的客户端请求已被批准，您必须查看添加到集群中的每台机器的服务器请求：

```
$ oc get csr
```

输出示例

```
NAME        AGE     REQUESTOR                                                   CONDITION
csr-bfd72   5m26s   system:node:ip-10-0-50-126.us-east-2.compute.internal
Pending
csr-c57lv   5m26s   system:node:ip-10-0-95-157.us-east-2.compute.internal
Pending
...
```

5. 如果剩余的 CSR 没有被批准，且处于 **Pending** 状态，请批准集群机器的 CSR：

- 若要单独批准，请对每个有效的 CSR 运行以下命令：

```
$ oc adm certificate approve <csr_name> 1
```

1 **<csr_name>** 是当前 CSR 列表中 CSR 的名称。

- 要批准所有待处理的 CSR，请运行以下命令：

```
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}{{"\n"}}
{{end}}{{end}}' | xargs oc adm certificate approve
```

6. 批准所有客户端和服务器 CSR 后，器将处于 **Ready** 状态。运行以下命令验证：

```
$ oc get nodes
```

输出示例

```
NAME      STATUS    ROLES   AGE  VERSION
master-0  Ready     master  73m  v1.20.0
master-1  Ready     master  73m  v1.20.0
master-2  Ready     master  74m  v1.20.0
worker-0  Ready     worker  11m  v1.20.0
worker-1  Ready     worker  11m  v1.20.0
```

注意

批准服务器 CSR 后可能需要几分钟时间让机器转换为 **Ready** 状态。

其他信息

- 如需有关 CSR 的更多信息，请参阅证书签名请求。

11.4.18. 初始 Operator 配置

在 control plane 初始化后，您必须立即配置一些 Operator 以便它们都可用。

先决条件

- 您的 control plane 已初始化。

流程

1. 观察集群组件上线：

```
$ watch -n5 oc get clusteroperators
```

输出示例

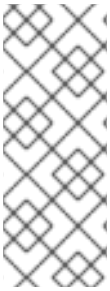
```
NAME                                       VERSION AVAILABLE   PROGRESSING   DEGRADED
SINCE
authentication                             4.6.0   True        False         False      3h56m
cloud-credential                           4.6.0   True        False         False      29h
cluster-autoscaler                         4.6.0   True        False         False      29h
config-operator                            4.6.0   True        False         False      6h39m
console                                    4.6.0   True        False         False      3h59m
csi-snapshot-controller                    4.6.0   True        False         False      4h12m
dns                                        4.6.0   True        False         False      4h15m
etcd                                       4.6.0   True        False         False      29h
image-registry                             4.6.0   True        False         False      3h59m
ingress                                    4.6.0   True        False         False      4h30m
insights                                   4.6.0   True        False         False      29h
kube-apiserver                             4.6.0   True        False         False      29h
kube-controller-manager                    4.6.0   True        False         False      29h
kube-scheduler                             4.6.0   True        False         False      29h
kube-storage-version-migrator              4.6.0   True        False         False      4h2m
machine-api                                4.6.0   True        False         False      29h
machine-approver                           4.6.0   True        False         False      6h34m
machine-config                             4.6.0   True        False         False      3h56m
marketplace                                4.6.0   True        False         False      4h2m
monitoring                                 4.6.0   True        False         False      6h31m
network                                    4.6.0   True        False         False      29h
node-tuning                                4.6.0   True        False         False      4h30m
openshift-apiserver                        4.6.0   True        False         False      3h56m
openshift-controller-manager               4.6.0   True        False         False      4h36m
openshift-samples                          4.6.0   True        False         False      4h30m
operator-lifecycle-manager                 4.6.0   True        False         False      29h
operator-lifecycle-manager-catalog         4.6.0   True        False         False      29h
operator-lifecycle-manager-packageserver   4.6.0   True        False         False      3h59m
service-ca                                 4.6.0   True        False         False      29h
storage                                    4.6.0   True        False         False      4h30m
```

2. 配置不可用的 Operator。

11.4.18.1. 安装过程中删除的镜像 registry

在不提供可共享对象存储的平台上，OpenShift Image Registry Operator bootstraps 本身的状态是 **Removed**。这允许 **openshift-installer** 在这些平台类型上完成安装。

将 **ManagementState** Image Registry Operator 配置从 **Removed** 改为 **Managed**。



注意

Prometheus 控制台提供了一个 **ImageRegistryRemoved** 警报，例如：

"Image Registry has been removed.**ImageStreamTags**, **BuildConfigs** and **DeploymentConfigs** which reference **ImageStreamTags** may not work as expected.Please configure storage and update the config to **Managed** state by editing configs.imageregistry.operator.openshift.io."

11.4.18.2. 镜像 registry 存储配置

对于不提供默认存储的平台，Image Registry Operator 最初将不可用。安装后，您必须配置 registry 使用的存储，这样 Registry Operator 才可用。

示配置生产集群所需的持久性卷的说明。如果适用，显示有关将空目录配置为存储位置的说明，该位置只可用于非生产集群。

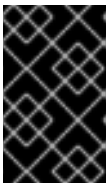
另外还提供了在升级过程中使用 **Recreate** rollout 策略来允许镜像 registry 使用块存储类型的说明。

11.4.18.2.1. 为 VMware vSphere 配置 registry 存储

作为集群管理员，在安装后需要配置 registry 来使用存储。

先决条件

- 具有 Cluster Administrator 权限
- VMware vSphere上有一个集群。
- 为集群置备的持久性存储，如 Red Hat OpenShift Container Storage。



重要

如果您只有一个副本，OpenShift Container Platform 支持对镜像 registry 存储的 **ReadWriteOnce** 访问。要部署支持高可用性的、带有两个或多个副本的镜像 registry，需要 **ReadWriteMany** 访问设置。

- 必须有“ 100Gi”容量。

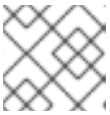
重要

测试显示，在 RHEL 中使用 NFS 服务器作为核心服务的存储后端可能会出现问题。这包括 OpenShift Container Registry 和 Quay，Prometheus 用于监控存储，以及 Elasticsearch 用于日志存储。因此，不推荐使用 RHEL NFS 作为 PV 后端用于核心服务。

市场上的其他 NFS 实现可能没有这些问题。如需了解更多与此问题相关的信息，请联络相关的 NFS 厂商。

流程

1. 为了配置 registry 使用存储，需要修改 **configs.imageregistry/cluster** 资源中的 **spec.storage.pvc** 。

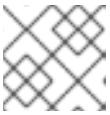


注意

使用共享存储时，请查看您的安全设置以防止被外部访问。

2. 验证您没有 registry pod:

```
$ oc get pod -n openshift-image-registry
```



注意

如果存储类型为 **emptyDIR**，则副本数不能超过 **1**。

3. 检查 registry 配置：

```
$ oc edit configs.imageregistry.operator.openshift.io
```

输出示例

```
storage:
  pvc:
    claim: 1
```

1 将 **claim** 字段留空以允许自动创建一个 **image-registry-storage** PVC。

4. 检查 **clusteroperator** 的状态：

```
$ oc get clusteroperator image-registry
```

11.4.18.2.2. 在非生产集群中配置镜像 registry 存储

您必须为 Image Registry Operator 配置存储。对于非生产集群，您可以将镜像 registry 设置为空目录。如果您这样做，重启 registry 后会丢失所有镜像。

流程

- 将镜像 registry 存储设置为空目录：

```
$ oc patch configs.imageregistry.operator.openshift.io cluster --type merge --patch '{"spec":{"storage":{"emptyDir":{}}}}'
```

警告

仅可为非生产集群配置这个选项。

如果在 Image Registry Operator 初始化其组件前运行此命令，**oc patch** 命令会失败并显示以下错误：

```
Error from server (NotFound): configs.imageregistry.operator.openshift.io "cluster" not found
```

等待几分钟，然后再次运行该命令。

11.4.18.2.3. 为 VMware vSphere 配置块 registry 存储

在作为集群管理员升级时，要允许镜像 registry 使用块存储类型，如 vSphere Virtual Machine Disk（VMDK），您可以使用 **Recreate** rollout 策略。

重要

支持块存储卷，但不建议将其用于生产环境中的镜像 registry。在块存储上配置 registry 的安装不具有高可用性，因为 registry 无法拥有多个副本。

流程

1. 要将镜像 registry 存储设置为块存储类型，对 registry 进行补丁，使其使用 **Recreate** rollout 策略，且仅使用 **1** 个副本运行：

```
$ oc patch config.imageregistry.operator.openshift.io/cluster --type=merge -p '{"spec":
{"rolloutStrategy":"Recreate","replicas":1}}'
```

2. 为块存储设备置备 PV，并为该卷创建 PVC。请求的块卷使用 ReadWriteOnce（RWO）访问模式。

   a. 创建包含以下内容的 **pvc.yaml** 文件以定义 VMware vSphere **PersistentVolumeClaim**：

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: image-registry-storage 1
  namespace: openshift-image-registry 2
spec:
  accessModes:
  - ReadWriteOnce 3
  resources:
    requests:
      storage: 100Gi 4
```

   1. 代表 **PersistentVolumeClaim** 对象的唯一名称。
   2. **PersistentVolumeClaim** 对象的命名空间，即 **openshift-image-registry**。
   3. 持久性卷声明的访问模式。使用 **ReadWriteOnce** 时，单个节点可以通过读写权限挂载这个卷。
   4. 持久性卷声明的大小。

   b. 从文件创建 **PersistentVolumeClaim** 对象：

```
$ oc create -f pvc.yaml -n openshift-image-registry
```

3. 编辑 registry 配置，使其可以正确引用 PVC:

```
$ oc edit config.imageregistry.operator.openshift.io -o yaml
```

输出示例

```
storage:
  pvc:
    claim: 1
```

1 通过创建自定义 PVC，您可以将 **claim** 字段留空以用于默认自动创建 **image-registry-storage** PVC。

有关配置 registry 存储以便引用正确的 PVC 的说明，请参阅为 vSphere 配置 registry。

11.4.19. 在用户置备的基础架构上完成安装

完成 Operator 配置后，可以在您提供的基础架构上完成集群安装。

先决条件

- 您的 control plane 已初始化。
- 已完成初始 Operator 配置。

流程

1. 使用以下命令确认所有集群组件都已在线：

```
$ watch -n5 oc get clusteroperators
```

输出示例

```
NAME                                       VERSION AVAILABLE   PROGRESSING   DEGRADED
SINCE
authentication                             4.6.0   True        False         False      3h56m
cloud-credential                           4.6.0   True        False         False      29h
cluster-autoscaler                         4.6.0   True        False         False      29h
config-operator                            4.6.0   True        False         False      6h39m
console                                    4.6.0   True        False         False      3h59m
csi-snapshot-controller                    4.6.0   True        False         False      4h12m
dns                                        4.6.0   True        False         False      4h15m
etcd                                       4.6.0   True        False         False      29h
image-registry                             4.6.0   True        False         False      3h59m
ingress                                    4.6.0   True        False         False      4h30m
insights                                   4.6.0   True        False         False      29h
kube-apiserver                             4.6.0   True        False         False      29h
kube-controller-manager                    4.6.0   True        False         False      29h
kube-scheduler                             4.6.0   True        False         False      29h
kube-storage-version-migrator              4.6.0   True        False         False      4h2m
```

```
machine-api                                4.6.0   True        False         False      29h
machine-approver                           4.6.0   True        False         False      6h34m
machine-config                             4.6.0   True        False         False      3h56m
marketplace                                4.6.0   True        False         False      4h2m
monitoring                                 4.6.0   True        False         False      6h31m
network                                    4.6.0   True        False         False      29h
node-tuning                                4.6.0   True        False         False      4h30m
openshift-apiserver                        4.6.0   True        False         False      3h56m
openshift-controller-manager               4.6.0   True        False         False      4h36m
openshift-samples                          4.6.0   True        False         False      4h30m
operator-lifecycle-manager                 4.6.0   True        False         False      29h
operator-lifecycle-manager-catalog         4.6.0   True        False         False      29h
operator-lifecycle-manager-packageserver   4.6.0   True        False         False      3h59m
service-ca                                 4.6.0   True        False         False      29h
storage                                    4.6.0   True        False         False      4h30m
```

或者，通过以下命令，如果所有集群都可用您会接到通知。它还检索并显示凭证：

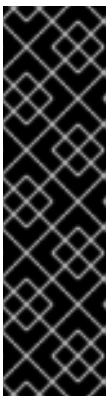
```
$ ./openshift-install --dir <installation_directory> wait-for install-complete 1
```

1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

输出示例

```
INFO Waiting up to 30m0s for the cluster to initialize...
```

Cluster Version Operator 完成从 Kubernetes API 服务器部署 OpenShift Container Platform 集群时，命令运行成功。



重要

- 安装程序生成的 Ignition 配置文件包含在 24 小时后过期的证书，然后在过期时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外情况是，您需要手动批准待处理的 **node-bootstrapper** 证书签名请求（CSR）来恢复 kubelet 证书。如需更多信息，请参阅*从过期的 control plane 证书中恢复*的文档。
- 建议您在生成 12 小时后使用 Ignition 配置文件，因为集群安装后 24 小时证书从 16 小时轮转至 22 小时。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中运行证书更新时避免安装失败。

2. 确认 Kubernetes API 服务器正在与 pod 通信。

   a. 要查看所有 pod 的列表，请使用以下命令：

   ```
   $ oc get pods --all-namespaces
   ```

   输出示例

   ```
   NAMESPACE                       NAME                                        READY   STATUS
   RESTARTS   AGE
   openshift-apiserver-operator    openshift-apiserver-operator-85cb746d55-zqhs8   1/1
   Running    1        9m
   ```

```
openshift-apiserver            apiserver-67b9g                          1/1     Running     0
3m
openshift-apiserver            apiserver-ljcmx                          1/1     Running     0
1m
openshift-apiserver            apiserver-z25h4                          1/1     Running     0
2m
openshift-authentication-operator authentication-operator-69d5d8bf84-vh2n8        1/1
Running     0          5m
...
```

b. 使用以下命令，查看上一命令的输出中所列 pod 的日志：

```
$ oc logs <pod_name> -n <namespace> 1
```

1 指定 pod 名称和命名空间，如上一命令的输出中所示。

如果 pod 日志显示，Kubernetes API 服务器可以与集群机器通信。

您可以按照将计算机器添加到 vSphere 的内容，在集群安装完成后添加额外的计算机器。

11.4.20. 备份 VMware vSphere 卷

OpenShift Container Platform 将新卷作为独立持久性磁盘置备，以便在集群中的任何节点上自由附加和分离卷。因此，无法备份使用快照的卷，也无法从快照中恢复卷。如需更多信息，请参阅 快照限制。

流程

要创建持久性卷的备份：

1. 停止使用持久性卷的应用程序。
2. 克隆持久性卷。
3. 重启应用程序。
4. 创建克隆的卷的备份。
5. 删除克隆的卷。

11.4.21. OpenShift Container Platform 的 Telemetry 访问

在 OpenShift Container Platform 4.6 中，默认运行的 Telemetry 服务提供有关集群健康状况和成功更新的指标，需要访问互联网。如果您的集群连接到互联网，Telemetry 会自动运行，而且集群会注册到 OpenShift Cluster Manager。

确认 OpenShift Cluster Manager 清单正确后，可以由 Telemetry 自动维护，也可以使用 OpenShift Cluster Manager 手动维护，使用订阅监控来跟踪帐户或多集群级别的 OpenShift Container Platform 订阅。

其他资源

- 有关 Telemetry 服务的更多信息，请参阅关于远程健康监控。

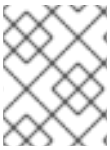
11.4.22. 后续步骤

- 自定义集群。
- 如果需要，您可以选择不使用远程健康报告。
- 设置 registry 并配置 registry 存储。

11.5. 使用网络自定义在 VSPHERE 上安装集群

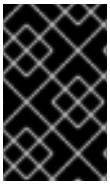
在 OpenShift Container Platform 版本 4.6 中，您可以使用自定义的网络配置选项在 VMware vSphere 环境中安装集群。通过自定义网络配置，您的集群可以与环境中现有的 IP 地址分配共存，并与现有的 MTU 和 VXLAN 配置集成。

大部分网络配置参数必须在安装过程中设置，只有 **kubeProxy** 配置参数可以在运行的集群中修改。



注意

OpenShift Container Platform 支持将集群部署到单个 VMware vCenter 中。不支持在多个 vCenter 上使用机器/机器集部署集群。



重要

进行用户置备的基础架构安装的步骤仅作为示例。使用您提供的基础架构安装集群需要了解 vSphere 平台和 OpenShift Container Platform 的安装过程。使用用户置备的基础架构安装说明作为指南；您可以通过其他方法创建所需的资源。

11.5.1. 先决条件

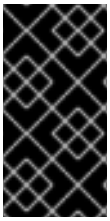
- 查看有关 OpenShift Container Platform 安装和更新流程的详细信息。
- 完成安装要求您在 vSphere 主机上上传 Red Hat Enterprise Linux CoreOS(RHCOS)OVA。完成此过程的机器需要访问 vCenter 和 ESXi 主机上的端口 443。验证是否可以访问端口 443。
- 如果您使用防火墙，您与管理员确认可以访问端口 443。control plane 节点必须能够通过端口 443 访问 vCenter 和 ESXi 主机，才能成功安装。
- 如果使用防火墙，您必须 将其配置为访问 Red Hat Insights。

11.5.2. OpenShift Container Platform 的互联网访问

在 OpenShift Container Platform 4.6 中，您需要访问互联网来安装集群。

您必须具有以下互联网访问权限：

- 访问 OpenShift Cluster Manager 以下载安装程序并执行订阅管理。如果集群可以访问互联网，并且没有禁用 Telemetry，该服务会自动授权您的集群。
- 访问 Quay.io，以获取安装集群所需的软件包。
- 获取执行集群更新所需的软件包。



重要

如果您的集群无法直接访问互联网，则可以在置备的某些类基础架构上执行受限网络安装。在此过程中，您要下载所需的内容，并使用它在镜像 registry（mirror registry） 中填充安装集群并生成安装程序所需的软件包。对于某些安装类型，集群要安装到的环境不需要访问互联网。在更新集群之前，要更新 registry 镜像系统中的内容。

11.5.3. VMware vSphere 基础架构要求

您必须在满足您使用的组件要求的 VMware vSphere 版本 6 或 7 实例上安装 OpenShift Container Platform 集群。

表 11.39. VMware 组件支持的最低 vSphere 版本

组件	最低支持版本	描述
虚拟机监控程序	vSphere 6.5 及之后的版本 13	此版本是 Red Hat Enterprise Linux CoreOS(RHCOS)支持的最低版本。请查看 Red Hat Enterprise Linux 8 支持的管理程序列表。
使用 in-tree 驱动程序存储	vSphere 6.5 及之后的版本	此插件使用 OpenShift Container Platform 中包含的 vSphere 的树内存储驱动程序创建 vSphere 存储。
可选： Networking (NSX-T)	vSphere 6.5U3 或 vSphere 6.7U2 及之后的版本	OpenShift Container Platform 需要 vSphere 6.5U3 或 vSphere 6.7U2+。VMware 的 NSX Container Plug-in（NCP）3.0.2 使用 OpenShift Container Platform 4.6 和 NSX-T 3.x+ 认证。

如果您使用 vSphere 版本 6.5 实例，请在安装 OpenShift Container Platform 前考虑升级到 6.7U3 或 7.0。

重要

您必须确保在安装 OpenShift Container Platform 前同步 ESXi 主机上的时间。请参阅 VMware 文档中的编辑主机时间配置。

11.5.4. 具有用户置备基础架构的集群的机器要求

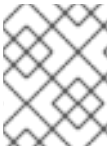
对于含有用户置备的基础架构的集群，您必须部署所有所需的机器。

11.5.4.1. 所需的机器

最小的 OpenShift Container Platform 集群需要下列主机：

- 一个临时 bootstrap 机器

- 三台 control plane 或 master 机器
- 至少两台计算机器，也称为 worker 机器。



注意

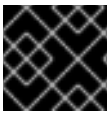
集群要求 bootstrap 机器在三台 control plane 机器上部署 OpenShift Container Platform 集群。您可在安装集群后删除 bootstrap 机器。

重要

要保持集群的高可用性，请将独立的物理主机用于这些集群机器。

bootstrap 和 control plane 机器必须使用 Red Hat Enterprise Linux CoreOS (RHCOS) 作为操作系统。但是，计算机器可以在 Red Hat Enterprise Linux CoreOS(RHCOS)或 Red Hat Enterprise Linux(RHEL)7.9 之间进行选择。

请注意，RHCOS 基于 Red Hat Enterprise Linux（RHEL） 8，并继承其所有硬件认证和要求。请查看Red Hat Enterprise Linux 技术功能及限制 。



重要

所有虚拟机必须位于与安装程序相同的数据存储中。

11.5.4.2. 网络连接要求

所有 Red Hat Enterprise Linux CoreOS (RHCOS) 机器在启动过程中需要 **initramfs** 中的网络从 Machine Config Server 获取 Ignition 配置文件。在初次启动过程中，需要一个 DHCP 服务器或设置了静态 IP 地址来建立网络连接，以下载它们的 Ignition 配置文件。另外，集群中的每个 OpenShift Container Platform 节点都必须有权访问网络时间协议（NTP）服务器。如果 DHCP 服务器提供 NTP 服务器信息，Red Hat Enterprise Linux CoreOS（RHCOS）机器上的 chrony 时间服务会读取信息，并可与 NTP 服务器同步时钟。

11.5.4.3. 最低资源要求

每台集群机器都必须满足以下最低要求：

表 11.40. 最低资源要求

机器	操作系统	vCPU [1]	虚拟内存	存储	IOPS [2]
bootstrap	RHCOS	4	16 GB	100 GB	300
Control plane	RHCOS	4	16 GB	100 GB	300
Compute	RHCOS 或 RHEL 7.9	2	8 GB	100 GB	300

1. 当未启用并发多线程 (SMT) 或超线程时，一个 vCPU 相当于一个物理内核。启用后，使用以下公式来计算对应的比例：（每个内核数的线程）× sockets = vCPU。

2. OpenShift Container Platform 和 Kubernetes 对磁盘性能非常敏感，建议使用更快的存储速度，特别是 control plane 节点上需要 10 ms p99 fsync 持续时间的 etcd。请注意，在许多云平台上，存储大小和 IOPS 可一起扩展，因此您可能需要过度分配存储卷来获取足够的性能。

11.5.4.4. 证书签名请求管理

在使用您置备的基础架构时，集群只能有限地访问自动机器管理，因此您必须提供一种在安装后批准集群证书签名请求 (CSR) 的机制。**kube-controller-manager** 只能批准 kubelet 客户端 CSR。 **machine-approver** 无法保证使用 kubelet 凭证请求的提供证书的有效性，因为它不能确认是正确的机器发出了该请求。您必须决定并实施一种方法，以验证 kubelet 提供证书请求的有效性并进行批准。

11.5.5. 创建用户置备的基础架构

在部署采用用户置备的基础架构的 OpenShift Container Platform 集群前，您必须创建底层基础架构。

先决条件

- 在为集群创建支持基础架构之前，请参阅OpenShift Container Platform 4.x Tested Integrations页。

流程

1. 在每个节点上配置 DHCP 或设置静态 IP 地址。
2. 提供所需的负载均衡器。
3. 配置机器的端口。
4. 配置 DNS。
5. 确保网络可以正常工作。

11.5.5.1. 用户置备的基础架构对网络的要求

所有 Red Hat Enterprise Linux CoreOS（RHCOS）机器在启动过程中需要 **initramfs** 中的网络从机器配置服务器获取 Ignition 配置。

在初次启动过程中，需要一个 DHCP 服务器或集群中的每个机器都设置了静态 IP 地址来建立网络连接，以下载它们的 Ignition 配置文件。

建议您使用 DHCP 服务器为集群进行长期机器管理。确保 DHCP 服务器已配置为向集群机器提供持久 IP 地址和主机名。

Kubernetes API 服务器必须能够解析集群机器的节点名称。如果 API 服务器和 worker 节点位于不同的区域中，您可以配置默认 DNS 搜索区域，以便 API 服务器能够解析节点名称。另一种支持的方法是始终在节点对象和所有 DNS 请求中使用完全限定域名来指代主机。

您必须配置机器间的网络连接，以便集群组件进行通信。每台机器都必须能够解析集群中所有其他机器的主机名。

表 11.41. 所有机器到所有机器

协议	端口	描述
ICMP	N/A	网络可访问性测试

协议	端口	描述
TCP	**1936**	指标
	9000-9999	主机级别的服务，包括端口 **9100-9101** 上的节点导出器和端口 **9099** 上的 Cluster Version Operator。
	10250-10259	Kubernetes 保留的默认端口
	10256	openshift-sdn
UDP	**4789**	VXLAN 和 Geneve
	6081	VXLAN 和 Geneve
	9000-9999	主机级别的服务，包括端口 **9100-9101** 上的节点导出器。
TCP/UDP	**30000-32767**	Kubernetes 节点端口

表 11.42. 要通过控制平面的所有机器

协议	端口	描述
TCP	**6443**	Kubernetes API

表 11.43. control plane 机器到 control plane 机器

协议	端口	描述
TCP	**2379-2380**	etcd 服务器和对等端口

网络拓扑要求
您为集群置备的基础架构必须满足下列网络拓扑要求。

重要

OpenShift Container Platform 要求所有节点都能访问互联网，以便为平台容器提取镜像并向红帽提供遥测数据。

负载均衡器
在安装 OpenShift Container Platform 前，您必须置备两个满足以下要求的负载均衡器：

1. **API 负载均衡器**：提供一个通用端点，供用户（包括人和机器）与平台交互和配置。配置以下条件：

- 只适用于第 4 层负载均衡。这可被称为 Raw TCP、SSL Passthrough 或者 SSL 桥接模式。如果使用 SSL Bridge 模式，必须为 API 路由启用 Server Name Indication（SNI）。
- 无状态负载平衡算法。这些选项根据负载均衡器的实现而有所不同。

重要

不要为 API 负载均衡器配置会话持久性。

在负载均衡器的前端和后台配置以下端口：

表 11.44. API 负载均衡器

端口	后端机器（池成员）	内部	外部	描述
6443	Bootstrap 和 control plane.bootstrap 机器初始化集群 control plane 后，您要从负载均衡器中删除 bootstrap 机器。您必须为 API 服务器健康检查探测配置 **/readyz** 端点。	X	X	Kubernetes API 服务器
22623	Bootstrap 和 control plane.bootstrap 机器初始化集群 control plane 后，您要从负载均衡器中删除 bootstrap 机器。	X		机器配置服务器

注意

负载均衡器必须配置为，从 API 服务器关闭 **/readyz** 端点到从池中删除 API 服务器实例时最多需要 30 秒。在 **/readyz** 返回错误或处于健康状态后的时间范围内，端点必须被删除或添加。每 5 秒或 10 秒探测一次，有两个成功请求处于健康状态，三个成为不健康的请求经过测试。

2. **应用程序入口负载均衡器**:提供来自集群外部的应用程序流量流量的 Ingress 点。配置以下条件：
   - 只适用于第 4 层负载均衡。这可被称为 Raw TCP、SSL Passthrough 或者 SSL 桥接模式。如果使用 SSL Bridge 模式，您必须为 Ingress 路由启用Server Name Indication（SNI）。
   - 建议根据可用选项以及平台上托管的应用程序类型，使用基于连接的或者基于会话的持久性。

在负载均衡器的前端和后台配置以下端口：

表 11.45. 应用程序入口负载均衡器

端口	后端机器（池成员）	内部	外部	描述
443	默认运行入口路由器 Pod、计算或 worker 的机器。	X	X	HTTPS 流量
80	默认运行入口路由器 Pod、计算或 worker 的机器。	X	X	HTTP 流量

提示

如果负载均衡器可以看到客户端的真实 IP 地址，启用基于 IP 的会话持久性可提高使用端到端 TLS 加密的应用程序的性能。

注意

OpenShift Container Platform 集群需要正确配置入口路由器。control plane 初始化后，您必须配置入口路由器。

以太网适配器硬件地址要求
当为集群置备虚拟机时，为每个虚拟机配置的以太网接口必须使用 VMware 机构唯一识别符 (OUI) 分配范围中的 MAC 地址：

- **00:05:69:00:00:00** 到 **00:05:69:FF:FF:FF**
- **00:0c:29:00:00:00** 到 **00:0c:29:FF:FF:FF**
- **00:1c:14:00:00:00** 到 **00:1c:14:FF:FF:FF**
- **00:50:56:00:00:00** 到 **00:50:56:FF:FF:FF**

如果使用 VMware OUI 以外的 MAC 地址，集群安装将无法成功。

NTP 配置
OpenShift Container Platform 集群默认配置为使用公共网络时间协议（NTP）服务器。如果要使用本地企业 NTP 服务器，或者集群部署在断开连接的网络中，您可以将集群配置为使用特定的时间服务器。如需更多信息，请参阅*配置 chrony 时间服务*的文档。

如果 DHCP 服务器提供 NTP 服务器信息，Red Hat Enterprise Linux CoreOS（RHCOS）机器上的 chrony 时间服务会读取信息，并可与 NTP 服务器同步时钟。

其他资源

- 配置 chrony 时间服务

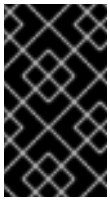
11.5.5.2. 用户置备 DNS 要求

DNS 用于名称解析和反向名称解析。DNS A/AAAA 或 CNAME 记录用于名称解析，PTR 记录用于反向解析名称。反向记录很重要，因为 Red Hat Enterprise Linux CoreOS（RHCOS）使用反向记录为所有节点设置主机名。另外，反向记录用于生成 OpenShift Container Platform 需要操作的证书签名请求（CSR）。

采用用户置备的基础架构的 OpenShift Container Platform 集群需要以下 DNS 记录。在每一记录中，**<cluster_name>** 是集群名称，**<base_domain>** 则是您在 **install-config.yaml** 文件中指定的集群基域。完整的 DNS 记录采用如下格式: **<component>.<cluster_name>.<base_domain>.**。

表 11.46. 所需的 DNS 记录

组件	记录	描述
Kubernetes API	**api.<cluster_name>.<base_domain>.**	添加 DNS A/AAAA 或 CNAME 记录，以及 DNS PTR 记录，以识别 control plane 机器的负载均衡器。这些记录必须由集群外的客户端以及集群中的所有节点解析。

组件	记录	描述
	api-int.<cluster_name>. <base_domain>.	添加 DNS A/AAAA 或 CNAME 记录，以及 DNS PTR 记录，以识别 control plane 机器的负载均衡器。这些记录必须可以从集群中的所有节点解析。  **重要** API 服务器必须能够根据在 Kubernetes 中记录的主机名解析 worker 节点。如果 API 服务器无法解析节点名称，则代理的 API 调用会失败，且您无法从 pod 检索日志。
Routes	***.apps.<cluster_name>. <base_domain>.**	添加通配符 DNS A/AAAA 或 CNAME 记录，指向以运行入口路由器 Pod 的机器（默认为 worker 节点）为目标的负载均衡器。这些记录必须由集群外的客户端以及集群中的所有节点解析。
bootstrap	**bootstrap.<cluster_name>. <base_domain>.**	添加 DNS A/AAAA 或 CNAME 记录，以及 DNS PTR 记录来识别 bootstrap 机器。这些记录必须由集群中的节点解析。
Master 主机	**<master><n>. <cluster_name>. <base_domain>.**	DNS A/AAAA 或 CNAME 记录，以识别 control plane 节点（也称为 master 节点）的每台机器。这些记录必须由集群中的节点解析。
Worker 主机	**<worker><n>. <cluster_name>. <base_domain>.**	添加 DNS A/AAAA 或 CNAME 记录，以识别 worker 节点的每台机器。这些记录必须由集群中的节点解析。

提示

您可以使用 **nslookup <hostname>** 命令来验证名称解析。您可以使用 **dig -x <ip_address>** 命令来验证 PTR 记录的反向名称解析。

下面的 BIND 区文件的例子展示了关于名字解析的 A 记录的例子。这个示例的目的是显示所需的记录。这个示例不是为选择一个名称解析服务提供建议。

例 11.9. DNS 区数据库示例

```
$TTL 1W
@ IN SOA ns1.example.com. root (
   2019070700 ; serial
   3H  ; refresh (3 hours)
   30M  ; retry (30 minutes)
   2W  ; expiry (2 weeks)
   1W )  ; minimum (1 week)
 IN NS ns1.example.com.
 IN MX 10 smtp.example.com.
;
;
```

```
ns1 IN A 192.168.1.5
smtp IN A 192.168.1.5
;
helper IN A 192.168.1.5
helper.ocp4 IN A 192.168.1.5
;
; The api identifies the IP of your load balancer.
api.ocp4  IN A 192.168.1.5
api-int.ocp4  IN A 192.168.1.5
;
; The wildcard also identifies the load balancer.
*.apps.ocp4  IN A 192.168.1.5
;
; Create an entry for the bootstrap host.
bootstrap.ocp4 IN A 192.168.1.96
;
; Create entries for the master hosts.
master0.ocp4  IN A 192.168.1.97
master1.ocp4  IN A 192.168.1.98
master2.ocp4  IN A 192.168.1.99
;
; Create entries for the worker hosts.
worker0.ocp4  IN A 192.168.1.11
worker1.ocp4  IN A 192.168.1.7
;
;EOF
```

下面的 BIND 区文件示例显示了反向名字解析的 PTR 记录示例。

例 11.10. 反向记录的 DNS 区数据库示例

```
$TTL 1W
@ IN SOA ns1.example.com. root (
   2019070700 ; serial
   3H  ; refresh (3 hours)
   30M  ; retry (30 minutes)
   2W  ; expiry (2 weeks)
   1W )  ; minimum (1 week)
 IN NS ns1.example.com.
;
; The syntax is "last octet" and the host must have an FQDN
; with a trailing dot.
97 IN PTR master0.ocp4.example.com.
98 IN PTR master1.ocp4.example.com.
99 IN PTR master2.ocp4.example.com.
;
96 IN PTR bootstrap.ocp4.example.com.
;
5 IN PTR api.ocp4.example.com.
5 IN PTR api-int.ocp4.example.com.
;
11 IN PTR worker0.ocp4.example.com.
```

```
7 IN PTR worker1.ocp4.example.com.
;
;EOF
```

11.5.6. 生成 SSH 私钥并将其添加到代理中

如果要在集群上执行安装调试或灾难恢复，则必须为 **ssh-agent** 和安装程序提供 SSH 密钥。您可以使用此密钥访问公共集群中的 bootstrap 机器来排除安装问题。

注意

在生产环境中，您需要进行灾难恢复和调试。

您可以使用此密钥以 **core** 用户身份通过 SSH 连接到 master 节点。在部署集群时，此密钥会添加到 **core** 用户的 **~/.ssh/authorized_keys** 列表中。

注意

您必须使用一个本地密钥，而不要使用在特定平台上配置的密钥，如 AWS 密钥对。

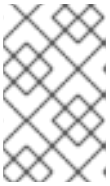
流程

1. 如果还没有为计算机上免密码身份验证而配置的 SSH 密钥，请创建一个。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ ssh-keygen -t ed25519 -N '' \
  -f <path>/<file_name> 1
```

1 指定新 SSH 密钥的路径和文件名，如 **~/.ssh/id_rsa**。如果您已有密钥对，请确保您的公钥位于 **~/.ssh** 目录中。

运行此命令会在指定的位置生成不需要密码的 SSH 密钥。



注意

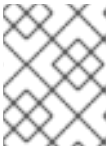
如果您计划在 **x86_64** 架构中安装使用 FIPS 验证的/Modules in Process 加密库的 OpenShift Container Platform 集群，不要创建使用 **ed25519** 算法的密钥。反之，创建一个使用 **rsa** 或 **ecdsa** 算法的密钥。

2. 作为后台任务启动 **ssh-agent** 进程：

```
$ eval "$(ssh-agent -s)"
```

输出示例

```
Agent pid 31874
```



注意

如果您的集群采用 FIPS 模式，则只使用 FIPS 兼容算法来生成 SSH 密钥。密钥必须是 RSA 或 ECDSA。

3. 将 SSH 私钥添加到 **ssh-agent**：

```
$ ssh-add <path>/<file_name> 1
```

输出示例

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

1 指定 SSH 私钥的路径和文件名，如 **~/.ssh/id_rsa**

后续步骤

- 在安装 OpenShift Container Platform 时，为安装程序提供 SSH 公钥。

11.5.7. 获取安装程序

在安装 OpenShift Container Platform 之前，将安装文件下载到本地计算机上。

先决条件

- 运行 Linux 或 macOS 的计算机，本地磁盘空间为 500 MB

流程

1. 访问 OpenShift Cluster Manager 站点的 Infrastructure Provider 页面。如果您有红帽帐号，请使用自己的凭证登录。如果没有，请创建一个帐户。
2. 选择您的基础架构供应商。
3. 进入适用于您的安装类型的页面，下载您的操作系统的安装程序，并将文件放在要保存安装配置文件的目录中。。

重要

安装程序会在用来安装集群的计算机上创建若干文件。在完成集群安装后，您必须保留安装程序和安装程序所创建的文件。这两个文件都需要删除集群。

重要

删除安装程序创建的文件不会删除您的集群，即使集群在安装过程中失败也是如此。要删除集群，为特定云供应商完成 OpenShift Container Platform 卸载流程。

4. 提取安装程序。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ tar xvf openshift-install-linux.tar.gz
```

5. 从 Red Hat OpenShift Cluster Manager 下载安装 pull secret 。通过此 pull secret，您可以进行所含授权机构提供的服务的身份验证，这些服务包括为 OpenShift Container Platform 组件提供容器镜像的 Quay.io。

11.5.8. 手动创建安装配置文件

对于使用用户置备的基础架构的 OpenShift Container Platform 安装，您必须手动生成安装配置文件。

先决条件

- 获取 OpenShift Container Platform 安装程序和集群的访问令牌。

流程

1. 创建用来存储您所需的安装资产的安装目录：

```
$ mkdir <installation_directory>
```

重要

您必须创建目录。一些安装信息，如 bootstrap X.509 证书，有较短的过期间隔，因此不要重复使用安装目录。如果要重复使用另一个集群安装中的个别文件，可以将其复制到您的目录中。但是，一些安装数据的文件名可能会在发行版本之间有所改变。从 OpenShift Container Platform 老版本中复制安装文件时要格外小心。

2. 自定义以下 **install-config.yaml** 文件模板，并将它保存到 **<installation_directory>** 中。

注意

此配置文件必须命名为 **install-config.yaml**。

3. 备份 **install-config.yaml** 文件，以便用于安装多个集群。

重要

install-config.yaml 文件会在安装过程的下一步骤中消耗掉。现在必须备份它。

11.5.8.1. VMware vSphere install-config.yaml 文件示例

您可以自定义 **install-config.yaml** 文件，以指定有关 OpenShift Container Platform 集群平台的更多信息，或修改所需参数的值。

```
apiVersion: v1
baseDomain: example.com (1)
compute:
- hyperthreading: Enabled (2) (3)
  name: worker
  replicas: 0 (4)
controlPlane:
  hyperthreading: Enabled (5) (6)
  name: master
  replicas: 3 (7)
```

```
metadata:
  name: test 8
platform:
  vsphere:
    vcenter: your.vcenter.server 9
    username: username 10
    password: password 11
    datacenter: datacenter 12
    defaultDatastore: datastore 13
    folder: "/<datacenter_name>/vm/<folder_name>/<subfolder_name>" 14
fips: false 15
pullSecret: '{"auths": ...}' 16
sshKey: 'ssh-ed25519 AAAA...' 17
```

1 集群的基域。所有 DNS 记录都必须是这个基域的子域，并包含集群名称。

2 5 **controlPlane** 部分是一个单映射，但 compute 部分是一系列映射。为满足不同数据结构的要求，**compute** 部分的第一行必须以连字符 **-** 开头，**controlPlane** 部分的第一行则不可以连字符开头。虽然这两个部分目前都定义单个机器池，但未来的 OpenShift Container Platform 版本可能会支持在安装过程中定义多个计算池。只使用一个 control plane 池。

3 6 是否要启用或禁用并发多线程或**超线程**。默认情况下，启用并发多线程以提高机器内核的性能。您可以通过将参数值设为 **Disabled** 来禁用。如果您在某些集群机器上禁用并发多线程，则必须在所有集群机器上禁用。

重要

如果禁用并发多线程，请确保在容量规划时考虑到机器性能可能会显著降低的问题。如果您禁用并发多线程，则计算机必须至少使用 8 个 CPU 和 32GB RAM。

4 **replicas** 参数的值必须设置为 **0**。此参数控制集群为您创建和管理的 worker 数量，使用用户置备的基础架构时集群不会执行这些功能。在完成 OpenShift Container Platform 安装前，您必须手动为集群部署 worker 机器。

7 您添加到集群的 control plane 机器数量。由于集群将这个值用作集群中 etcd 端点的数量，因此该值必须与您部署的 control plane 机器数量匹配。

8 您在 DNS 记录中指定的集群名称。

9 vCenter 服务器的完全限定主机名或 IP 地址。

10 用于访问服务器的用户名。此用户必须至少具有 vSphere 中 静态或动态持久性卷置备 所需的角色和权限。

11 与 vSphere 用户关联的密码。

12 vSphere 数据中心。

13 要使用的默认 vSphere 数据存储。

14 可选：对于安装程序置备的基础架构，安装程序创建虚拟机的现有文件夹的绝对路径，如 **/<datacenter_name>/vm/<folder_name>/<subfolder_name>**。如果没有提供这个值，安装程序会在数据中心虚拟机文件夹中创建一个顶层文件夹，其名称为基础架构 ID。如果您为集群提供基础架构,请省略此参数。

15 是否启用或禁用 FIPS 模式。默认情况下不启用 FIPS 模式。如果启用了 FIPS 模式，运行 OpenShift Container Platform 的 Red Hat Enterprise Linux CoreOS (RHCOS) 机器会绕过默认的 Kubernetes

重要

只有在 **x86_64** 架构中的 OpenShift Container Platform 部署支持 FIPS 验证的/Modules in Process 加密库。

16 从 OpenShift Cluster Manager 获取的 pull secret。通过此 pull secret，您可以进行所含授权机构提供的服务的身份验证，这些服务包括为 OpenShift Container Platform 组件提供容器镜像的 Quay.io。

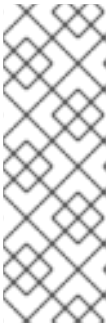
17 Red Hat Enterprise Linux CoreOS (RHCOS) 中 **core** 用户的默认 SSH 密钥的公钥部分。

11.5.8.2. 在安装过程中配置集群范围代理

生产环境可能会拒绝直接访问互联网，而是提供 HTTP 或 HTTPS 代理。您可以通过在 **install-config.yaml** 文件中配置代理设置，将新的 OpenShift Container Platform 集群配置为使用代理。

先决条件

- 您有一个现有的 **install-config.yaml** 文件。
- 您检查了集群需要访问的站点，并决定是否需要绕过代理。默认情况下代理所有集群出口流量，包括对托管云供应商 API 的调用。您需要将站点添加到 **Proxy** 对象的 **spec.noProxy** 字段来绕过代理。



注意

Proxy 对象 **status.noProxy** 字段使用安装配置中的 **networking.machineNetwork[].cidr**、**networking.clusterNetwork[].cidr** 和 **networking.serviceNetwork[]** 字段的值填充。

对于在 Amazon Web Services(AWS)、Google Cloud Platform(GCP)、Microsoft Azure 和 Red Hat OpenStack Platform(RHOSP)上安装, **Proxy** 对象 **status.noProxy** 字段也会使用实例元数据端点填充(**169.254.169.254**)。

流程

1. 编辑 **install-config.yaml** 文件并添加代理设置。例如：

```
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: example.com 3
additionalTrustBundle: | 4
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
...
```

1. 用于创建集群外 HTTP 连接的代理 URL。URL 必须是 **http**。
2. 用于创建集群外 HTTPS 连接的代理 URL。
3. 要排除在代理中的目标域名、IP 地址或其他网络 CIDR 的逗号分隔列表。在域前面加 **.** 来仅匹配子域。例如： **.y.com** 匹配 **x.y.com**，但不匹配 **y.com**。使用 ***** 绕过所有目的地的代理。您必须包含 vCenter 的 IP 地址以及用于其机器的 IP 范围。
4. 如果提供，安装程序会在 **openshift-config 命名空间中生成名为 user-ca- bundle** 的配置映射来保存额外的 CA 证书。如果您提供 **additionalTrustBundle** 和至少一个代理设置，则 **Proxy** 对象会被配置为引用 **trustedCA** 字段中的 **user-ca-bundle** 配置映射。然后，Cluster Network Operator 会创建一个 **trusted-ca-bundle** 配置映射，该配置映射将为 **trustedCA** 参数指定的内容与 RHCOS 信任捆绑包合并。**additionalTrustBundle** 字段是必需的，除非代理的身份证书由来自 RHCOS 信任捆绑包的颁发机构签名。

注意

安装程序不支持代理的 **readinessEndpoints** 字段。

2. 保存该文件，并在安装 OpenShift Container Platform 时引用。

安装程序会创建一个名为 **cluster** 的集群范围代理，该代理使用提供的 **install-config.yaml** 文件中的代理设置。如果没有提供代理设置，仍然会创建一个 **cluster Proxy** 对象，但它会有一个空 **spec**。

注意

只支持名为 **cluster** 的 **Proxy** 对象，且无法创建额外的代理。

11.5.9. 网络配置阶段

当在安装前指定集群配置时，在安装过程中的几个阶段可以修改网络配置：

阶段 1

输入 **openshift-install create install-config** 命令后。在 **install-config.yaml** 文件中，您可以自定义以下与网络相关的字段：

- **networking.networkType**
- **networking.clusterNetwork**
- **networking.serviceNetwork**
- **networking.machineNetwork**
  有关这些字段的更多信息，请参阅"安装配置参数"。

注意

将 **networking.machineNetwork** 设置为与首选 NIC 所在的 CIDR 匹配。

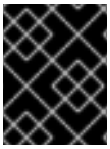
阶段 2

输入 **openshift-install create manifests** 命令后。如果必须指定高级网络配置，在这个阶段中，只能使用您要修改的字段来定义自定义的 Cluster Network Operator 清单。

在 2 阶段，您无法覆盖 **install-config.yaml** 文件中的 1 阶段中指定的值。但是，您可以在第 2 阶段进一步自定义集群网络供应商。

11.5.10. 指定高级网络配置

您可以通过为集群网络供应商指定额外的配置，使用高级配置自定义将集群整合到现有网络环境中。您只能在安装集群前指定高级网络配置。



重要

不支持修改安装程序创建的 OpenShift Container Platform 清单文件。支持应用您创建的清单文件，如以下流程所示。

先决条件

- 创建 **install-config.yaml** 文件并完成对其所做的任何修改。
- 为集群生成 Ignition 配置文件。

流程

1. 进入包含安装程序的目录并创建清单：

```
$ ./openshift-install create manifests --dir <installation_directory>
```

其中：

<installation_directory>
指定包含集群的 **install-config.yaml** 文件的目录名称。

2. 在 **<installation_directory>/manifests/** 目录下，为高级网络配置创建一个名为 **cluster-network-03-config.yml** 的 stub 清单文件：

```
$ cat <<EOF > <installation_directory>/manifests/cluster-network-03-config.yml
apiVersion: operator.openshift.io/v1
kind: Network
metadata:
  name: cluster
spec:
EOF
```

其中：

<installation_directory>
指定包含集群的 **manifests/** 目录的目录名称。

3. 在编辑器中打开 **cluster-network-03-config.yml** 文件，并为集群指定高级网络配置，如下例所示：

为 OpenShift SDN 网络供应商指定不同的 VXLAN 端口

```
apiVersion: operator.openshift.io/v1
kind: Network
metadata:
```

```
  name: cluster
spec:
  defaultNetwork:
    openshiftSDNConfig:
      vxlanPort: 4800
```

4. 保存 **cluster-network-03-config.yml** 文件，再退出文本编辑器。

5. 可选：备份 **manifests/cluster-network-03-config.yml** 文件。创建集群时，安装程序会删除 **manifests/** 目录。

6. 删除定义 control plane 机器的 Kubernetes 清单文件以及计算 machineSets：

```
$ rm -f openshift/99_openshift-cluster-api_master-machines-*.yaml openshift/99_openshift-cluster-api_worker-machineset-*.yaml
```

由于您要自行创建和管理这些资源，因此不必初始化这些资源。

- 您可以使用机器 API 来保留 MachineSet 文件来创建计算机器，但您必须更新对其的引用，以匹配您的环境。

11.5.11. Cluster Network Operator 配置

集群网络的配置作为 Cluster Network Operator (CNO) 配置的一部分被指定，并存储在名为 **cluster**的自定义资源（CR）对象中。CR 指定 **operator.openshift.io** API 组中的 **Network** API 的字段。

CNO 配置会在集群安装过程中从 **Network.config.openshift.io** API 组中的 **Network** API 继承以下字段，这些字段无法更改：

clusterNetwork

从中分配 pod IP 地址的 IP 地址池。

serviceNetwork

服务的 IP 地址池。

defaultNetwork.type

集群网络供应商，如 OpenShift SDN 或 OVN-Kubernetes。

您可以通过在名为 **cluster** 的 CNO 对象中设置 **defaultNetwork** 对象的字段来为集群指定集群网络供应商配置。

11.5.11.1. Cluster Network Operator 配置对象

Cluster Network Operator（CNO）的字段在下表中描述：

表 11.47. Cluster Network Operator 配置对象

字段	类型	描述
metadata.name	**字符串**	CNO 对象的名称。这个名称始终是 **cluster**。

字段	类型	描述
spec.clusterNetwork	**数组**	用于指定从哪些 IP 地址块分配 Pod IP 地址以及分配给集群中每个节点的子网前缀长度的列表。例如： spec: clusterNetwork: - cidr: 10.128.0.0/19 hostPrefix: 23 - cidr: 10.128.32.0/19 hostPrefix: 23 此值是只读的，并在 **install-config.yaml** 文件中指定。
spec.serviceNetwork	**数组**	服务的 IP 地址块。OpenShift SDN 和 OVN-Kubernetes Container Network Interface（CNI）网络供应商只支持服务网络具有单个 IP 地址块。例如： spec: serviceNetwork: - 172.30.0.0/14 此值是只读的，并在 **install-config.yaml** 文件中指定。
spec.defaultNetwork	**对象**	为集群网络配置 Container Network Interface（CNI）集群网络供应商。
spec.kubeProxyConfig	**对象**	此对象的字段指定 kube-proxy 配置。如果您使用 OVN-Kubernetes 集群网络供应商，则 kube-proxy 的配置不会起作用。

defaultNetwork 对象配置

defaultNetwork 对象的值在下表中定义：

表 11.48. defaultNetwork 对象

字段	类型	描述
type	**字符串**	**OpenShiftSDN** 或 **OVNKubernetes**。在安装过程中选择了集群网络供应商。集群安装后无法更改这个值。  **注意** OpenShift Container Platform 默认使用 OpenShift SDN Container Network Interface（CNI）集群网络供应商。

字段	类型	描述
openshiftSDNConfig	**对象**	此对象仅对 OpenShift SDN 集群网络供应商有效。
ovnKubernetesConfig	**对象**	此对象仅对 OVN-Kubernetes 集群网络供应商有效。

配置 OpenShift SDN CNI 集群网络供应商

下表描述了 OpenShift SDN Container Network Interface（CNI）集群网络供应商的配置字段。

表 11.49. openshiftSDNConfig 对象

字段	类型	描述
mode	**字符串**	配置 OpenShift SDN 的网络隔离模式。默认值为 **NetworkPolicy**。 **Multitenant** 和 **Subnet** 的值可以向后兼容 OpenShift Container Platform 3.x，但不推荐这样做。集群安装后无法更改这个值。
mtu	**整数**	VXLAN 覆盖网络的最大传输单元 (MTU) 。这根据主网络接口的 MTU 自动探测。您通常不需要覆盖检测到的 MTU。 如果自动探测的值不是您期望的，请确认节点上主网络接口中的 MTU 是正确的。您不能使用这个选项更改节点上主网络接口的 MTU 值。 如果您的集群中的不同节点需要不同的 MTU 值，则必须将此值设置为比集群中的最低 MTU 值小 **50**。例如，如果集群中的某些节点的 MTU 为 **9001**，而某些节点的 MTU 为 **1500**，则必须将此值设置为 **1450**。 集群安装后无法更改这个值。
vxlanPort	**整数**	用于所有 VXLAN 数据包的端口。默认值为 **4789**。集群安装后无法更改这个值。 如果您在虚拟环境中运行，并且现有节点是另一个 VXLAN 网络的一部分，那么可能需要更改此值。例如，当在 VMware NSX-T 上运行 OpenShift SDN 覆盖时，您必须为 VXLAN 选择一个备用端口，因为两个 SDN 都使用相同的默认 VXLAN 端口号。 在 Amazon Web Services (AWS) 上，您可以在端口 **9000** 和端口 **9999** 之间为 VXLAN 选择一个备用端口。

OpenShift SDN 配置示例

```
defaultNetwork:
  type: OpenShiftSDN
  openshiftSDNConfig:
```

```
    mode: NetworkPolicy
    mtu: 1450
    vxlanPort: 4789
```

配置 OVN-Kubernetes CNI 集群网络供应商
下表描述了 OVN-Kubernetes CNI 集群网络供应商的配置字段。

表 11.50. ovnKubernetesConfig 对象

字段	类型	描述
mtu	**整数**	Geneve（Generic Network Virtualization Encapsulation）覆盖网络的最大传输单元（MTU）。这根据主网络接口的 MTU 自动探测。您通常不需要覆盖检测到的 MTU。 如果自动探测的值不是您期望的，请确认节点上主网络接口中的 MTU 是正确的。您不能使用这个选项更改节点上主网络接口的 MTU 值。 如果您的集群中的不同节点需要不同的 MTU 值，则必须将此值设置为比集群中的最低 MTU 值小 **100**。例如，如果集群中的某些节点的 MTU 为 **9001**，而某些节点的 MTU 为 **1500**，则必须将此值设置为 **1400**。 集群安装后无法更改这个值。
genevePort	**整数**	用于所有 Geneve 数据包的端口。默认值为 **6081**。集群安装后无法更改这个值。

OVN-Kubernetes 配置示例

```
defaultNetwork:
  type: OVNKubernetes
  ovnKubernetesConfig:
    mtu: 1400
    genevePort: 6081
```

kubeProxyConfig 对象配置
kubeProxyConfig 对象的值在下表中定义：

表 11.51. kubeProxyConfig 对象

字段	类型	描述

字段	类型	描述
iptablesSyncPeriod	**字符串**	**iptables** 规则的刷新周期。默认值为 **30s**。有效的后缀包括 **s**、**m** 和 **h**，具体参见 Go **time** 软件包文档。  **注意** 由于 OpenShift Container Platform 4.3 及更高版本中引进了性能上的改进，现在不再需要调整 **iptablesSyncPeriod** 参数。

proxyArguments.iptables-min-sync-period	**数组**	刷新 **iptables** 规则前的最短时长。此字段确保刷新的频率不会过于频繁。有效的后缀包括 **s**、**m** 和 **h**，具体参见 Go **time** 软件包。默认值为：
		
		`kubeProxyConfig:`
		`  proxyArguments:`
		`  iptables-min-sync-period:`
		`  - 0s`

11.5.12. 创建 Ignition 配置文件

由于需要手工启动集群机器，因此您必须生成 Ignition 配置文件，集群需要它来创建其机器。

重要

- 安装程序生成的 Ignition 配置文件包含在 24 小时后过期的证书，然后在过期时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外情况是，您需要手动批准待处理的 **node-bootstrapper** 证书签名请求（CSR）来恢复 kubelet 证书。如需更多信息，请参阅 *从过期的 control plane 证书中恢复* 的文档。
- 建议您在生成 12 小时后使用 Ignition 配置文件，因为集群安装后 24 小时证书从 16 小时轮转至 22 小时。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中运行证书更新时避免安装失败。

先决条件

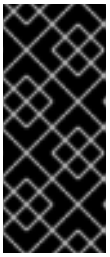
- 获取 OpenShift Container Platform 安装程序以及集群的 pull secret。

流程

- 获取 Ignition 配置文件：

```
$ ./openshift-install create ignition-configs --dir <installation_directory> 1
```

1 对于 **<installation_directory>**，请指定用于保存安装程序所创建的文件的目录名称。



重要

如果您创建了 **install-config.yaml** 文件，请指定包含该文件的目录。否则，指定一个空目录。一些安装信息，如 bootstrap X.509 证书，有较短的过期间隔，因此不要重复使用安装目录。如果要重复使用另一个集群安装中的个别文件，可以将其复制到您的目录中。但是，一些安装数据的文件名可能会在发行版本之间有所改变。从 OpenShift Container Platform 老版本中复制安装文件时要格外小心。

该目录中将生成以下文件：

```
.
├── auth
│   ├── kubeadmin-password
│   └── kubeconfig
├── bootstrap.ign
├── master.ign
├── metadata.json
└── worker.ign
```

11.5.13. 提取基础架构名称

Ignition 配置文件包含一个唯一的集群标识符,您可以使用它在 VMware vSphere 中唯一地标识您的集群。如果计划使用集群标识符作为虚拟机文件夹的名称,您必须提取它。

先决条件

- 获取 OpenShift Container Platform 安装程序以及集群的 pull secret。
- 已为集群生成 Ignition 配置文件。
- 安装了 **jq** 软件包。

流程

- 要从 Ignition 配置文件元数据中提取和查看基础架构名称，请运行以下命令：

```
$ jq -r .infraID <installation_directory>/metadata.json 1
```

1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

输出示例

```
openshift-vw9j6 1
```

1 此命令的输出是您的集群名称和随机字符串。

11.5.14. 在 vSphere 中创建 Red Hat Enterprise Linux CoreOS (RHCOS) 机器

在 VMware vSphere 上安装包含用户置备基础架构的集群前，您必须在 vSphere 主机上创建 RHCOS 机器供其使用。

先决条件

- 已获取集群的 Ignition 配置文件。
- 您可以从计算机访问 HTTP 服务器，并且您创建的机器也可以访问该服务器。
- 您已创建了 vSphere 集群。

流程

1. 将名为 **<installation_directory>/bootstrap.ign** 的 bootstrap Ignition 配置文件上传到 HTTP 服务器，该配置文件是由安装程序创建的。记下此文件的 URL。
2. 将 bootstrap 节点的以下辅助 Ignition 配置文件保存到计算机中，存为 **<installation_directory>/merge-bootstrap.ign** ：

```
{
  "ignition": {
    "config": {
      "merge": [
        {
          "source": "<bootstrap_ignition_config_url>", 1
          "verification": {}
        }
      ]
    },
    "timeouts": {},
    "version": "3.1.0"
  },
  "networkd": {},
  "passwd": {},
  "storage": {},
  "systemd": {}
}
```

1 指定您托管的 bootstrap Ignition 配置文件的 URL。

为 bootstrap 机器创建虚拟机 (VM) 时，您要使用此 Ignition 配置文件。

3. 找到安装程序创建的以下 Ignition 配置文件：
   - **<installation_directory>/master.ign**
   - **<installation_directory>/worker.ign**
   - **<installation_directory>/merge-bootstrap.ign**
4. 将 Ignition 配置文件转换为 Base64 编码。在此流程中，您必须将这些文件添加到虚拟机中的额外配置参数 **guestinfo.ignition.config.data** 中。
   例如，如果您使用 Linux 操作系统，可以使用 **base64** 命令来编码这些文件。

```
$ base64 -w0 <installation_directory>/master.ign > <installation_directory>/master.64
```

```
$ base64 -w0 <installation_directory>/worker.ign > <installation_directory>/worker.64
```

```
$ base64 -w0 <installation_directory>/merge-bootstrap.ign > <installation_directory>/merge-bootstrap.64
```

重要

如果您计划在安装完成后在集群中添加更多计算机器，请不要删除这些文件。

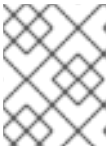
5. 获取 RHCOS OVA 镜像。镜像位于 RHCOS 镜像镜像页面。

重要

RHCOS 镜像可能不会随着 OpenShift Container Platform 的每一发行版本都有改变。您必须下载一个最高版本的镜像，其版本号应小于或等于您安装的 OpenShift Container Platform 版本。如果可用，请使用与 OpenShift Container Platform 版本匹配的镜像版本。

文件名包含 OpenShift Container Platform 版本号，格式为 **rhcos-vmware.<architecture>.ova**。

6. 在 vSphere 客户端中，在数据中心中创建一个文件夹来存储您的虚拟机。
   a. 点击 **VMs and Templates** 视图。
   b. 右键点击您的数据中心名称。
   c. 点击 **New Folder → New VM and Template Folder**。
   d. 在显示的窗口中输入文件夹名称。如果您没有在 **install-config.yaml** 文件中指定现有文件夹，请创建一个文件夹，其名称与基础架构 ID 相同。您可以使用这个文件夹名称，因此 vCenter 会在适当的位置为 Workspace 配置动态置备存储。
7. 在 vSphere 客户端中，为 OVA 镜像创建一个模板，然后根据需要克隆模板。



注意

在以下步骤中，您将创建一个模板，然后克隆所有集群机器的模板。然后，在置备虚拟机时，为该克隆的机器类型提供 Ignition 配置文件的位置。

   a. 在 **Hosts and Clusters** 选项卡中，右键点击您的集群名称并选择 **Deploy OVF Template**。
   b. 在 **Select an OVF** 选项卡中，指定您下载的 RHCOS OVA 文件的名称。
   c. 在 **Select a name and folder** 选项卡中，为您的模板设置 **虚拟机名称**，如 **Template-RHCOS**。点击 vSphere 集群的名称并选择您在上一步中创建的文件夹。
   d. 在 **Select a compute resource** 选项卡中，点击您的 vSphere 集群名称。
   e. 在 **Select storage** 选项卡中，配置虚拟机的存储选项。
      - 根据您的存储要求，选择**Thin Provision** 或 **Thick Provision**。
      - 选择您在 **install-config.yaml** 文件中指定的数据存储。
   f. 在 **Select network** 选项卡中，指定您为集群配置的网络（如果可用）。

g. 在创建 OVF 模板时，请不要在 **Customize template** 选项卡上指定值，或者不要再配置模板。

重要

不要启动原始虚拟机模板。VM 模板必须保持关闭状态，必须为新的 RHCOS 机器克隆。启动虚拟机模板会将虚拟机模板配置为平台上的虚拟机，这样可防止它被用作计算机集可以应用配置的模板。

8. 部署模板后，为集群中的机器部署虚拟机。

a. 右键点击模板的名称，再点击 **Clone → Clone to Virtual Machine**。

b. 在 **Select a name and folder**选项卡中，指定虚拟机的名称。名称中可以包括机器类型，如 **control-plane-0** 或 **compute-1**。

c. 在 **Select a name and folder**选项卡中，选择您为集群创建的文件夹名称。

d. 在 **Select a compute resource** 选项卡中，选择数据中心中的主机名称。
对于 bootstrap 机器，指定您托管的 bootstrap Ignition 配置文件的 URL。

e. 可选：在 **Select storage** 选项卡中，自定义存储选项。

f. 在 **Select clone options** 中，选择 **Customize this virtual machine's hardware**。

g. 在 **Customize hardware** 选项卡中，点击 **VM Options → Advanced**。

- 可选：覆盖 vSphere 中的默认 DHCP 网络。启用静态 IP 网络：

i. 设置静态 IP 配置：

```
$ export IPCFG="ip=<ip>::<gateway>:<netmask>:<hostname>:<iface>:none
nameserver=srv1 [nameserver=srv2 [nameserver=srv3 [...]]]"
```

示例命令

```
$ export IPCFG="ip=192.168.100.101::192.168.100.254:255.255.255.0:::none
nameserver=8.8.8.8"
```

ii. 在从 vSphere 中的 OVA 引导虚拟机前，设置 **guestinfo.afterburn.initrd.network-kargs** 属性：

```
$ govc vm.change -vm "<vm_name>" -e "guestinfo.afterburn.initrd.network-
kargs=${IPCFG}"
```

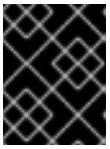
- 可选：在出现集群性能问题时，从 **Latency Sensitivity** 列表中选择 **High**。确定虚拟机的 CPU 和内存保留有以下值：
  - 内存保留值必须等于其配置的内存大小。
  - CPU 保留值必须至少是低延迟虚拟 CPU 的数量，乘以测量的物理 CPU 速度。
- 点击 **Edit Configuration**，然后在 **Configuration Parameters** 窗口中点击 **Add Configuration Params**。定义以下参数名称和值：

- **guestinfo.ignition.config.data** ：找到您在此流程中创建的 base-64 编码文件，并粘贴此机器类型的 base64 编码 Ignition 配置文件的内容。
- **guestinfo.ignition.config.data.encoding**：指定 **base64**。
- **disk.EnableUUID**：指定 **TRUE**。

h. 在 **Customize hardware** 选项卡的 **Virtual Hardware** 面板中，根据需要修改指定的值。确保 RAM、CPU 和磁盘存储的数量满足机器类型的最低要求。

i. 完成配置并打开虚拟机电源。

9. 对于每台机器，按照前面的步骤为集群创建其余的机器。



重要

此刻您必须创建 bootstrap 和 control plane 机器。由于计算机器中已默认部署了一些 Pod，因此在安装集群前，还要创建至少两台计算机器。

11.5.15. 在 vSphere 中创建更多 Red Hat Enterprise Linux CoreOS（RHCOS）机器

您可以为集群创建更多计算机器，在 VMware vSphere 上使用用户置备的基础架构。

先决条件

- 获取计算机器的 Base64 编码 Ignition 文件。
- 您可以访问您为集群创建的 vSphere 模板。

流程

1. 部署模板后，为集群中的机器部署虚拟机。

a. 右键点击模板的名称，再点击 **Clone → Clone to Virtual Machine**。

b. 在 **Select a name and folder**选项卡中，指定虚拟机的名称。您可以在名称中包含机器类型，如 **compute-1**。

c. 在 **Select a name and folder**选项卡中，选择您为集群创建的文件夹名称。

d. 在 **Select a compute resource** 选项卡中，选择数据中心中的主机名称。

e. 可选：在 **Select storage** 选项卡中，自定义存储选项。

f. 在 **Select clone options** 中，选择 **Customize this virtual machine's hardware**。

g. 在 **Customize hardware** 选项卡中，点击 **VM Options → Advanced**。

- 从 **Latency Sensitivity** 列表中选择 **High**。
- 点击 **Edit Configuration**，然后在 **Configuration Parameters** 窗口中点击 **Add Configuration Params**。定义以下参数名称和值：
  - **guestinfo.ignition.config.data**：粘贴此机器类型的 Base64 编码计算 Ignition 配置文件的内容。
  - **guestinfo.ignition.config.data.encoding**：指定 **base64**。

- **disk.EnableUUID**：指定 **TRUE**。

h. 在 **Customize hardware** 选项卡的 **Virtual Hardware** 面板中，根据需要修改指定的值。确保 RAM、CPU 和磁盘存储的数量满足机器类型的最低要求。另外，如果有多个可用的网络，请确定在 **Add network adapter 中** 选择正确的网络。

i. 完成配置并打开虚拟机电源。

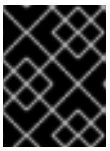
2. 继续为集群创建更多计算机器。

11.5.16. 磁盘分区

在大多数情况下，数据分区最初是由安装 RHCOS 而不是安装另一个操作系统来创建的。在这种情况下，OpenShift Container Platform 安装程序应该被允许配置磁盘分区。

但是，在安装 OpenShift Container Platform 节点时，在两种情况下您可能需要覆盖默认分区：

- 创建单独的分区：对于在空磁盘中的 greenfield 安装，您可能想要在分区中添加单独的存储。这只在生成 **/var** 或者一个 **/var** 独立分区的子目录（如 **/var/lib/etcd** ）时被正式支持，但不支持两者。



重要

Kubernetes 只支持两个文件系统分区。如果您在原始配置中添加多个分区，Kubernetes 无法监控所有这些分区。

- 保留现有分区：对于 brownfield 安装，您要在现有节点上重新安装 OpenShift Container Platform，并希望保留从之前的操作系统中安装的数据分区，对于 **coreos-installer** 来说，引导选项和选项都允许您保留现有数据分区。

创建一个独立的 /var 分区

通常情况下，OpenShift Container Platform 的磁盘分区应该留给安装程序。然而，在有些情况下您可能需要在文件系统的一部分中创建独立分区。

OpenShift Container Platform 支持添加单个分区来将存储附加到 **/var** 分区或 **/var** 的子目录。例如：

- **/var/lib/containers**：保存镜像相关的内容，随着更多镜像和容器添加到系统中，它所占用的存储会增加。
- **/var/lib/etcd**：保存您可能希望保持独立的数据，比如 etcd 存储的性能优化。
- **/var**：保存您希望独立保留的数据，用于特定目的（如审计）。

单独存储 **/var** 目录的内容可方便地根据需要对区域扩展存储，并可以在以后重新安装 OpenShift Container Platform 时保持该数据地完整。使用这个方法，您不必再次拉取所有容器，在更新系统时也无法复制大量日志文件。

因为 **/var** 在进行一个全新的 Red Hat Enterprise Linux CoreOS（RHCOS）安装前必需存在，所以这个流程会在 OpenShift Container Platform 安装过程的 **openshift-install** 准备阶段插入的机器配置来设置独立的 **/var** 分区。

流程

1. 创建存放 OpenShift Container Platform 安装文件的目录：

```
$ mkdir $HOME/clusterconfig
```

2. 运行 **openshift-install** 在 **manifest** 和 **openshift** 子目录中创建一组文件。在出现提示时回答系统问题：

```
$ openshift-install create manifests --dir $HOME/clusterconfig
? SSH Public Key ...
$ ls $HOME/clusterconfig/openshift/
99_kubeadmin-password-secret.yaml
99_openshift-cluster-api_master-machines-0.yaml
99_openshift-cluster-api_master-machines-1.yaml
99_openshift-cluster-api_master-machines-2.yaml
...
```

3. 创建 **MachineConfig** 对象并将其添加到 **openshift** 目录中的一个文件中。例如，把文件命名为 **98-var-partition.yaml**，将磁盘设备名称改为 **worker** 系统中存储设备的名称，并根据情况设置存储大小。这个示例将 **/var** 目录放在一个单独的分区中：

```
apiVersion: machineconfiguration.openshift.io/v1
kind: MachineConfig
metadata:
  labels:
    machineconfiguration.openshift.io/role: worker
  name: 98-var-partition
spec:
  config:
    ignition:
      version: 3.1.0
    storage:
      disks:
      - device: /dev/<device_name> 1
        partitions:
        - label: var
          startMiB: <partition_start_offset> 2
          sizeMiB: <partition_size> 3
      filesystems:
        - device: /dev/disk/by-partlabel/var
          path: /var
          format: xfs
    systemd:
      units:
        - name: var.mount 4
          enabled: true
          contents: |
            [Unit]
            Before=local-fs.target
            [Mount]
            What=/dev/disk/by-partlabel/var
            Where=/var
            Options=defaults,prjquota 5
            [Install]
            WantedBy=local-fs.target
```

1 要分区的磁盘的存储设备名称。

2 当在引导磁盘中添加数据分区时，推荐最少使用 25000MB。root 文件系统会自动重新定义大小使其占据所有可用空间（最多到指定的偏移值）。如果没有指定值，或者指定的值小于推荐的最小值，则生成的 root 文件系统会太小，而在以后进行的 RHCOS 重新安装可能会覆盖数据分区的开始部分。

3 数据分区的大小（以兆字节为单位）。

4 挂载单元的名称必须与 **Where=** 指令中指定的目录匹配。例如，对于挂载在 **/var/lib/containers** 上的文件系统，该单元必须命名为 **var-lib-containers.mount**。

5 对于用于容器存储的文件系统，必须启用 **prjquota** 挂载选项。

注意

在创建独立 **/var** 分区时，如果不同的实例类型没有相同的设备名称，则无法将不同的实例类型用于 worker 节点。

4. 再次运行 **openshift-install**，从 **manifest** 和 **openshift** 子目录中的一组文件创建 Ignition 配置：

```
$ openshift-install create ignition-configs --dir $HOME/clusterconfig
$ ls $HOME/clusterconfig/
auth  bootstrap.ign  master.ign  metadata.json  worker.ign
```

现在，您可以使用 Ignition 配置文件作为 vSphere 安装程序的输入来安装 Red Hat Enterprise Linux CoreOS（RHCOS）系统。

11.5.17. 创建集群

要创建 OpenShift Container Platform 集群，请等待您通过安装程序生成的 Ignition 配置文件所置备的机器上完成 bootstrap 过程。

先决条件

- 为集群创建所需的基础架构。
- 已获得安装程序并为集群生成了 Ignition 配置文件。
- 已使用 Ignition 配置文件为集群创建 RHCOS 机器。
- 您的机器可直接访问互联网，或者可以使用 HTTP 或 HTTPS 代理。

流程

1. 监控 bootstrap 过程：

```
$ ./openshift-install --dir <installation_directory> wait-for bootstrap-complete \ 1
    --log-level=info 2
```

1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

2 要查看不同的安装详情，请指定 **warn**、**debug** 或 **error**，而不要指定 **info**。

输出示例

```
INFO Waiting up to 30m0s for the Kubernetes API at https://api.test.example.com:6443...
INFO API v1.19.0 up
INFO Waiting up to 30m0s for bootstrapping to complete...
INFO It is now safe to remove the bootstrap resources
```

Kubernetes API 服务器提示已在 control plane 机器上完成 bootstrap 时，命令运行成功。

2. bootstrap 过程完成后，请从负载均衡器中删除 bootstrap 机器。

重要

此时您必须从负载均衡器中删除 bootstrap 机器。您还可以删除或重新格式化机器本身。

11.5.18. 使用 CLI 登录到集群

您可以通过导出集群 **kubeconfig** 文件，以默认系统用户身份登录集群。**kubeconfig** 文件包含关于集群的信息，供 CLI 用于将客户端连接到正确集群和 API 服务器。该文件特只适用于一个特定的集群，在 OpenShift Container Platform 安装过程中创建。

先决条件

- 已部署了 OpenShift Container Platform 集群。
- 已安装 **oc** CLI。

流程

1. 导出 **kubeadmin** 凭证：

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

2. 使用导出的配置，验证能否成功运行 **oc** 命令：

```
$ oc whoami
```

输出示例

```
system:admin
```

11.5.19. 批准机器的证书签名请求

将机器添加到集群时，会为您添加的每台机器生成两个待处理证书签名请求（CSR）。您必须确认这些 CSR 已获得批准，或根据需要自行批准。客户端请求必须首先被批准，然后是服务器请求。

先决条件

- 您已将机器添加到集群中。

流程

1. 确认集群可以识别这些机器：

```
$ oc get nodes
```

输出示例

```
NAME     STATUS   ROLES   AGE  VERSION
master-0 Ready    master  63m  v1.19.0
master-1 Ready    master  63m  v1.19.0
master-2 Ready    master  64m  v1.19.0
```

输出将列出您创建的所有机器。

注意

在一些 CSR 被批准前，以上输出可能不包括计算节点（也称为 worker 节点）。

2. 检查待处理的 CSR，并确保可以看到添加到集群中的每台机器都有 **Pending** 或 **Approved** 状态的客户端请求：

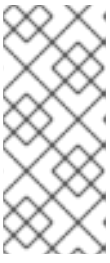
```
$ oc get csr
```

输出示例

```
NAME        AGE    REQUESTOR                                                    CONDITION
csr-8b2br   15m    system:serviceaccount:openshift-machine-config-operator:node-
bootstrapper   Pending
csr-8vnps   15m    system:serviceaccount:openshift-machine-config-operator:node-
bootstrapper   Pending
...
```

在本例中，两台机器加入了集群。您可能会在列表中看到更多已批准的 CSR。

3. 如果 CSR 没有获得批准，请在所添加机器的所有待处理 CSR 都处于 **Pending** 状态后，为您的集群机器批准这些 CSR：



注意

由于 CSR 会自动轮转，因此请在将机器添加到集群后一小时内批准您的 CSR。如果没有在一小时内批准，证书将会轮转，每个节点将会存在多个证书。您必须批准所有这些证书。批准客户端 CSR 后， Kubelet 为服务证书创建辅助 CSR，这需要手动批准。然后，如果 Kubelet 请求具有相同参数的新证书，则 **machine-approver** 会自动批准后续服务证书续订请求。

注意

对于在未启用机器 API 的平台中运行的集群，如裸机和其他用户置备的基础架构，必须采用一种方法自动批准 kubelet 提供证书请求（CSR）。如果没有批准请求，则 **oc exec**、**oc rsh** 和 **oc logs** 命令将无法成功，因为 API 服务器连接到 kubelet 时需要服务证书。与 Kubelet 端点联系的任何操作都需要此证书批准。这个方法必须监视新的 CSR，确认 CSR 由 **system:node** 或 **system:admin** 组中的 **node-bootstrapper** 服务帐户提交，并确认节点的身份。

- 若要单独批准，请对每个有效的 CSR 运行以下命令：

```
$ oc adm certificate approve <csr_name> 1
```

1 **<csr_name>** 是当前 CSR 列表中 CSR 的名称。

- 要批准所有待处理的 CSR，请运行以下命令：

```
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}{{"\n"}}
{{end}}{{end}}' | xargs --no-run-if-empty oc adm certificate approve
```

注意

在有些 CSR 被批准前，一些 Operator 可能无法使用。

4. 现在，您的客户端请求已被批准，您必须查看添加到集群中的每台机器的服务器请求：

```
$ oc get csr
```

输出示例

```
NAME        AGE     REQUESTOR                                                   CONDITION
csr-bfd72   5m26s   system:node:ip-10-0-50-126.us-east-2.compute.internal       Pending
csr-c57lv   5m26s   system:node:ip-10-0-95-157.us-east-2.compute.internal       Pending
...
```

5. 如果剩余的 CSR 没有被批准，且处于 **Pending** 状态，请批准集群机器的 CSR：

- 若要单独批准，请对每个有效的 CSR 运行以下命令：

```
$ oc adm certificate approve <csr_name> 1
```

1 **<csr_name>** 是当前 CSR 列表中 CSR 的名称。

- 要批准所有待处理的 CSR，请运行以下命令：

```
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}{{"\n"}}
{{end}}{{end}}' | xargs oc adm certificate approve
```

6. 批准所有客户端和服务器 CSR 后，器将处于 **Ready** 状态。运行以下命令验证：

```
$ oc get nodes
```

输出示例

```
NAME     STATUS   ROLES   AGE  VERSION
master-0 Ready    master  73m  v1.20.0
master-1 Ready    master  73m  v1.20.0
master-2 Ready    master  74m  v1.20.0
worker-0 Ready    worker  11m  v1.20.0
worker-1 Ready    worker  11m  v1.20.0
```

注意

批准服务器 CSR 后可能需要几分钟时间让机器转换为 **Ready** 状态。

其他信息

- 如需有关 CSR 的更多信息，请参阅证书签名请求。

11.5.19.1. 初始 Operator 配置

在 control plane 初始化后，您必须立即配置一些 Operator 以便它们都可用。

先决条件

- 您的 control plane 已初始化。

流程

1. 观察集群组件上线：

```
$ watch -n5 oc get clusteroperators
```

输出示例

```
NAME                                       VERSION AVAILABLE   PROGRESSING   DEGRADED
SINCE
authentication                             4.6.0   True        False         False      3h56m
cloud-credential                           4.6.0   True        False         False      29h
cluster-autoscaler                         4.6.0   True        False         False      29h
config-operator                            4.6.0   True        False         False      6h39m
console                                    4.6.0   True        False         False      3h59m
csi-snapshot-controller                    4.6.0   True        False         False      4h12m
dns                                        4.6.0   True        False         False      4h15m
etcd                                       4.6.0   True        False         False      29h
image-registry                             4.6.0   True        False         False      3h59m
ingress                                    4.6.0   True        False         False      4h30m
insights                                   4.6.0   True        False         False      29h
kube-apiserver                             4.6.0   True        False         False      29h
kube-controller-manager                    4.6.0   True        False         False      29h
kube-scheduler                             4.6.0   True        False         False      29h
kube-storage-version-migrator              4.6.0   True        False         False      4h2m
machine-api                                4.6.0   True        False         False      29h
```

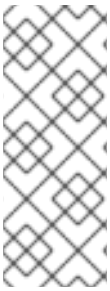
```
machine-approver                           4.6.0   True        False         False      6h34m
machine-config                             4.6.0   True        False         False      3h56m
marketplace                                4.6.0   True        False         False      4h2m
monitoring                                 4.6.0   True        False         False      6h31m
network                                    4.6.0   True        False         False      29h
node-tuning                                4.6.0   True        False         False      4h30m
openshift-apiserver                        4.6.0   True        False         False      3h56m
openshift-controller-manager               4.6.0   True        False         False      4h36m
openshift-samples                          4.6.0   True        False         False      4h30m
operator-lifecycle-manager                 4.6.0   True        False         False      29h
operator-lifecycle-manager-catalog         4.6.0   True        False         False      29h
operator-lifecycle-manager-packageserver   4.6.0   True        False         False      3h59m
service-ca                                 4.6.0   True        False         False      29h
storage                                    4.6.0   True        False         False      4h30m
```

2. 配置不可用的 Operator。

11.5.19.2. 安装过程中删除的镜像 registry

在不提供可共享对象存储的平台上，OpenShift Image Registry Operator bootstraps 本身的状态是 **Removed**。这允许 **openshift-installer** 在这些平台类型上完成安装。

将 **ManagementState** Image Registry Operator 配置从 **Removed** 改为 **Managed**。



注意

Prometheus 控制台提供了一个 **ImageRegistryRemoved** 警报，例如：

"Image Registry has been removed.**ImageStreamTags**, **BuildConfigs** and **DeploymentConfigs** which reference **ImageStreamTags** may not work as expected.Please configure storage and update the config to **Managed** state by editing configs.imageregistry.operator.openshift.io."

11.5.19.3. 镜像 registry 存储配置

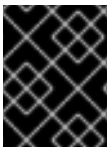
对于不提供默认存储的平台，Image Registry Operator 最初将不可用。安装后，您必须配置 registry 使用的存储，这样 Registry Operator 才可用。

示配置生产集群所需的持久性卷的说明。如果适用，显示有关将空目录配置为存储位置的说明，该位置只可用于非生产集群。

另外还提供了在升级过程中使用 **Recreate** rollout 策略来允许镜像 registry 使用块存储类型的说明。

11.5.19.3.1. 为 VMware vSphere 配置块 registry 存储

在作为集群管理员升级时，要允许镜像 registry 使用块存储类型，如 vSphere Virtual Machine Disk（VMDK），您可以使用 **Recreate** rollout 策略。



重要

支持块存储卷，但不建议将其用于生产环境中的镜像 registry。在块存储上配置 registry 的安装不具有高可用性，因为 registry 无法拥有多个副本。

流程

1. 要将镜像 registry 存储设置为块存储类型，对 registry 进行补丁，使其使用 **Recreate** rollout 策略，且仅使用 **1** 个副本运行：

```
$ oc patch config.imageregistry.operator.openshift.io/cluster --type=merge -p '{"spec":
{"rolloutStrategy":"Recreate","replicas":1}}'
```

2. 为块存储设备置备 PV，并为该卷创建 PVC。请求的块卷使用 ReadWriteOnce（RWO）访问模式。

   a. 创建包含以下内容的 **pvc.yaml** 文件以定义 VMware vSphere **PersistentVolumeClaim**：

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: image-registry-storage (1)
  namespace: openshift-image-registry (2)
spec:
  accessModes:
  - ReadWriteOnce (3)
  resources:
    requests:
      storage: 100Gi (4)
```

   (1) 代表 **PersistentVolumeClaim** 对象的唯一名称。

   (2) **PersistentVolumeClaim** 对象的命名空间，即 **openshift-image-registry**。

   (3) 持久性卷声明的访问模式。使用 **ReadWriteOnce** 时，单个节点可以通过读写权限挂载这个卷。

   (4) 持久性卷声明的大小。

   b. 从文件创建 **PersistentVolumeClaim** 对象：

```
$ oc create -f pvc.yaml -n openshift-image-registry
```

3. 编辑 registry 配置，使其可以正确引用 PVC:

```
$ oc edit config.imageregistry.operator.openshift.io -o yaml
```

输出示例

```
storage:
  pvc:
    claim: (1)
```

(1) 通过创建自定义 PVC，您可以将 **claim** 字段留空以用于默认自动创建 **image-registry-storage** PVC。

有关配置 registry 存储以便引用正确的 PVC 的说明，请参阅为 vSphere 配置 registry。

11.5.20. 在用户置备的基础架构上完成安装

完成 Operator 配置后，可以在您提供的基础架构上完成集群安装。

先决条件

- 您的 control plane 已初始化。
- 已完成初始 Operator 配置。

流程

1. 使用以下命令确认所有集群组件都已在线：

```
$ watch -n5 oc get clusteroperators
```

输出示例

```
NAME                                       VERSION AVAILABLE   PROGRESSING   DEGRADED
SINCE
authentication                             4.6.0   True        False         False      3h56m
cloud-credential                           4.6.0   True        False         False      29h
cluster-autoscaler                         4.6.0   True        False         False      29h
config-operator                            4.6.0   True        False         False      6h39m
console                                    4.6.0   True        False         False      3h59m
csi-snapshot-controller                    4.6.0   True        False         False      4h12m
dns                                        4.6.0   True        False         False      4h15m
etcd                                       4.6.0   True        False         False      29h
image-registry                             4.6.0   True        False         False      3h59m
ingress                                    4.6.0   True        False         False      4h30m
insights                                   4.6.0   True        False         False      29h
kube-apiserver                             4.6.0   True        False         False      29h
kube-controller-manager                    4.6.0   True        False         False      29h
kube-scheduler                             4.6.0   True        False         False      29h
kube-storage-version-migrator              4.6.0   True        False         False      4h2m
machine-api                                4.6.0   True        False         False      29h
machine-approver                           4.6.0   True        False         False      6h34m
machine-config                             4.6.0   True        False         False      3h56m
marketplace                                4.6.0   True        False         False      4h2m
monitoring                                 4.6.0   True        False         False      6h31m
network                                    4.6.0   True        False         False      29h
node-tuning                                4.6.0   True        False         False      4h30m
openshift-apiserver                        4.6.0   True        False         False      3h56m
openshift-controller-manager               4.6.0   True        False         False      4h36m
openshift-samples                          4.6.0   True        False         False      4h30m
operator-lifecycle-manager                 4.6.0   True        False         False      29h
operator-lifecycle-manager-catalog         4.6.0   True        False         False      29h
operator-lifecycle-manager-packageserver   4.6.0   True        False         False      3h59m
service-ca                                 4.6.0   True        False         False      29h
storage                                    4.6.0   True        False         False      4h30m
```

或者，通过以下命令，如果所有集群都可用您会接到通知。它还检索并显示凭证：

```
$ ./openshift-install --dir <installation_directory> wait-for install-complete 1
```

1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

输出示例

```
INFO Waiting up to 30m0s for the cluster to initialize...
```

Cluster Version Operator 完成从 Kubernetes API 服务器部署 OpenShift Container Platform 集群时，命令运行成功。

重要

- 安装程序生成的 Ignition 配置文件包含在 24 小时后过期的证书，然后在过期时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外情况是，您需要手动批准待处理的 **node-bootstrapper** 证书签名请求（CSR）来恢复 kubelet 证书。如需更多信息，请参阅*从过期的 control plane 证书中恢复*的文档。
- 建议您在生成 12 小时后使用 Ignition 配置文件，因为集群安装后 24 小时证书从 16 小时轮转至 22 小时。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中运行证书更新时避免安装失败。

2. 确认 Kubernetes API 服务器正在与 pod 通信。

   a. 要查看所有 pod 的列表，请使用以下命令：

   ```
   $ oc get pods --all-namespaces
   ```

   输出示例

   ```
   NAMESPACE                  NAME                                        READY   STATUS
   RESTARTS   AGE
   openshift-apiserver-operator    openshift-apiserver-operator-85cb746d55-zqhs8   1/1
   Running    1          9m
   openshift-apiserver        apiserver-67b9g                             1/1     Running    0
   3m
   openshift-apiserver        apiserver-ljcmx                             1/1     Running    0
   1m
   openshift-apiserver        apiserver-z25h4                             1/1     Running    0
   2m
   openshift-authentication-operator authentication-operator-69d5d8bf84-vh2n8       1/1
   Running    0          5m
   ...
   ```

   b. 使用以下命令，查看上一命令的输出中所列 pod 的日志：

   ```
   $ oc logs <pod_name> -n <namespace>
   ```
   1

   1 指定 pod 名称和命名空间，如上一命令的输出中所示。

   如果 pod 日志显示，Kubernetes API 服务器可以与集群机器通信。

您可以按照将计算机器添加到 vSphere 的内容，在集群安装完成后添加额外的计算机器。

11.5.21. 备份 VMware vSphere 卷

OpenShift Container Platform 将新卷作为独立持久性磁盘置备，以便在集群中的任何节点上自由附加和分离卷。因此，无法备份使用快照的卷，也无法从快照中恢复卷。如需更多信息，请参阅 快照限制。

流程

要创建持久性卷的备份：

1. 停止使用持久性卷的应用程序。
2. 克隆持久性卷。
3. 重启应用程序。
4. 创建克隆的卷的备份。
5. 删除克隆的卷。

11.5.22. OpenShift Container Platform 的 Telemetry 访问

在 OpenShift Container Platform 4.6 中，默认运行的 Telemetry 服务提供有关集群健康状况和成功更新的指标，需要访问互联网。如果您的集群连接到互联网，Telemetry 会自动运行，而且集群会注册到 OpenShift Cluster Manager。

确认 OpenShift Cluster Manager 清单正确后，可以由 Telemetry 自动维护，也可以使用 OpenShift Cluster Manager 手动维护，使用订阅监控来跟踪帐户或多集群级别的 OpenShift Container Platform 订阅。

其他资源

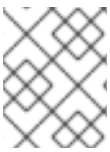
- 有关 Telemetry 服务的更多信息，请参阅关于远程健康监控。

11.5.23. 后续步骤

- 自定义集群。
- 如果需要，您可以选择不使用远程健康报告。
- 设置 registry 并配置 registry 存储。

11.6. 在受限网络中的 VSPHERE 上安装集群

在 OpenShift Container Platform 4.6 中，您可以通过创建安装发行内容的内部镜像在受限网络中的 VMware vSphere 基础架构上安装集群。



注意

OpenShift Container Platform 支持将集群部署到单个 VMware vCenter 中。不支持在多个 vCenter 上使用机器/机器集部署集群。

11.6.1. 先决条件

- 在镜像主机上创建镜像 registry，并获取您的 OpenShift Container Platform 版本的 **imageContentSources** 数据。

重要

由于安装介质位于堡垒主机上，因此请使用该计算机完成所有安装步骤。

- 为集群置备持久性存储。若要部署私有镜像 registry，您的存储必须提供 ReadWriteMany 访问模式。
- 查看有关 OpenShift Container Platform 安装和更新流程的详细信息。
- OpenShift Container Platform 安装程序需要访问 vCenter 和 ESXi 主机上的端口 443。您确认可以访问端口 443。
- 如果您使用防火墙，您与管理员确认可以访问端口 443。control plane 节点必须能够通过端口 443 访问 vCenter 和 ESXi 主机，才能成功安装。
- 如果使用防火墙并计划使用遥测（telemetry），您必须将防火墙配置为允许集群需要访问的站点。

注意

如果要配置代理，请务必还要查看此站点列表。

11.6.2. 关于在受限网络中安装

在 OpenShift Container Platform 4.6 中，可以执行不需要有效的互联网连接来获取软件组件的安装。受限网络安装可使用安装程序置备的基础架构或用户置备的基础架构完成，具体取决于您要安装集群的云平台。

如果选择在云平台中执行受限网络安装，仍然需要访问其云 API。有些云功能，比如 Amazon Web Service 的 Route 53 DNS 和 IAM 服务，需要访问互联网。根据您的网络，在裸机硬件或 VMware vSphere 上安装时可能需要较少的互联网访问。

要完成受限网络安装，您必须创建一个 registry，镜像 OpenShift Container Platform registry 的内容并包含其安装介质。您可以在堡垒主机上创建此镜像，该主机可同时访问互联网和您的封闭网络，也可以使用满足您的限制条件的其他方法。

11.6.2.1. 其他限制

受限网络中的集群还有以下额外限制：

- **ClusterVersion** 状态包含一个 **Unable to retrieve available updates** 错误。
- 默认情况下，您无法使用 Developer Catalog 的内容，因为您无法访问所需的镜像流标签。

11.6.3. OpenShift Container Platform 的互联网访问

在 OpenShift Container Platform 4.6 中，您需要访问互联网来获得用来安装集群的镜像。

您必须具有以下互联网访问权限：

- 访问 OpenShift Cluster Manager 以下载安装程序并执行订阅管理。如果集群可以访问互联网，并且没有禁用 Telemetry，该服务会自动授权您的集群。
- 访问 Quay.io，以获取安装集群所需的软件包。

- 获取执行集群更新所需的软件包。

重要

如果您的集群无法直接访问互联网，则可以在置备的某些类基础架构上执行受限网络安装。在此过程中，您要下载所需的内容，并使用它在镜像 registry（mirror registry） 中填充安装集群并生成安装程序所需的软件包。对于某些安装类型，集群要安装到的环境不需要访问互联网。在更新集群之前，要更新 registry 镜像系统中的内容。

11.6.4. VMware vSphere 基础架构要求

您必须在满足您使用的组件要求的 VMware vSphere 版本 6 或 7 实例上安装 OpenShift Container Platform 集群。

表 11.52. VMware 组件支持的最低 vSphere 版本

组件	最低支持版本	描述
虚拟机监控程序	vSphere 6.5 及之后的版本 13	此版本是 Red Hat Enterprise Linux CoreOS(RHCOS)支持的最低版本。请查看 Red Hat Enterprise Linux 8 支持的管理程序列表。
使用 in-tree 驱动程序存储	vSphere 6.5 及之后的版本	此插件使用 OpenShift Container Platform 中包含的 vSphere 的树内存储驱动程序创建 vSphere 存储。
可选： Networking (NSX-T)	vSphere 6.5U3 或 vSphere 6.7U2 及之后的版本	OpenShift Container Platform 需要 vSphere 6.5U3 或 vSphere 6.7U2+。VMware 的 NSX Container Plug-in（NCP）3.0.2 使用 OpenShift Container Platform 4.6 和 NSX-T 3.x+ 认证。

如果您使用 vSphere 版本 6.5 实例，请在安装 OpenShift Container Platform 前考虑升级到 6.7U3 或 7.0。

重要

您必须确保在安装 OpenShift Container Platform 前同步 ESXi 主机上的时间。请参阅 VMware 文档中的编辑主机时间配置。

11.6.5. 网络连接要求

您必须配置机器之间的网络连接，以允许 OpenShift Container Platform 集群组件进行通信。

查看有关所需网络端口的以下详细信息。

表 11.53. 用于全机器到所有机器通信的端口

协议	端口	描述
ICMP	N/A	网络可访问性测试
TCP	**1936**	指标
	9000-9999	主机级别的服务，包括端口 **9100-9101** 上的节点导出器和端口 **9099** 上的 Cluster Version Operator。
	10250-10259	Kubernetes 保留的默认端口
	10256	openshift-sdn
UDP	**4789**	虚拟可扩展 LAN(VXLAN)
	6081	Geneve
	9000-9999	主机级别的服务，包括端口 **9100-9101** 上的节点导出器。
	500	IPsec IKE 数据包
	4500	IPsec NAT-T 数据包
TCP/UDP	**30000-32767**	Kubernetes 节点端口
ESP	N/A	IPsec Encapsulating Security Payload(ESP)

表 11.54. 用于所有机器控制平面通信的端口

协议	端口	描述
TCP	**6443**	Kubernetes API

表 11.55. control plane 机器用于 control plane 机器通信的端口

协议	端口	描述
TCP	**2379-2380**	etcd 服务器和对等端口

11.6.6. vCenter 要求

在使用安装程序置备的基础架构的 vCenter 上安装 OpenShift Container Platform 集群前，您必须准备自己的环境。

所需的 vCenter 帐户权限
要在 vCenter 中安装 OpenShift Container Platform 集群，安装程序需要一个具有特权的帐户来读取和创建所需资源。使用具有全局管理特权的帐户是访问所有必要权限的最简单方式。

如果无法使用具有全局管理特权的帐户，您必须创建角色来授予 OpenShift Container Platform 集群安装所需的权限。虽然大多数权限始终是必需的，但是一些权限只有在计划安装程序需要在您的 vCenter 实例中置备一个包含 OpenShift Container Platform 集群的文件夹时（这是默认行为）才需要。您必须为指定对象创建或修改 vSphere 角色，才能授予所需的权限。

如果安装程序创建 vSphere 虚拟机文件夹，则需要额外的角色。

例 11.11. 安装所需的角色和权限

角色的 vSphere 对象	何时需要	所需的权限
vSphere vCenter	Always	**Cns.Searchable InventoryService.Tagging.AttachTag InventoryService.Tagging.CreateCategory InventoryService.Tagging.CreateTag InventoryService.Tagging.DeleteCategory InventoryService.Tagging.DeleteTag InventoryService.Tagging.EditCategory InventoryService.Tagging.EditTag Sessions.ValidateSession StorageProfile.View**
vSphere vCenter Cluster	Always	**Host.Config.Storage Resource.AssignVMToPool VApp.AssignResourcePool VApp.Import VirtualMachine.Config.AddNewDisk**
vSphere Datastore	Always	**Datastore.AllocateSpace Datastore.Browse Datastore.FileManagement**
vSphere 端口组	Always	**Network.Assign**

角色的 vSphere 对象	何时需要	所需的权限
虚拟机文件夹	Always	**Resource.AssignVMToPool VApp.Import VirtualMachine.Config.Add ExistingDisk VirtualMachine.Config.Add NewDisk VirtualMachine.Config.Add RemoveDevice VirtualMachine.Config.Adva ncedConfig VirtualMachine.Config.Anno tation VirtualMachine.Config.CPU Count VirtualMachine.Config.Disk Extend VirtualMachine.Config.Disk Lease VirtualMachine.Config.Edit Device VirtualMachine.Config.Mem ory VirtualMachine.Config.Rem oveDisk VirtualMachine.Config.Rena me VirtualMachine.Config.Rese tGuestInfo VirtualMachine.Config.Reso urce VirtualMachine.Config.Setti ngs VirtualMachine.Config.Upgr adeVirtualHardware VirtualMachine.Interact.Gue stControl VirtualMachine.Interact.Pow erOff VirtualMachine.Interact.Pow erOn VirtualMachine.Interact.Res et VirtualMachine.Inventory.Cr eate VirtualMachine.Inventory.Cr eateFromExisting VirtualMachine.Inventory.D elete VirtualMachine.Provisionin g.Clone**

角色的 vSphere 对象	何时需要	所需的权限
vSphere vCenter Datacenter	如果安装程序创建虚拟机文件夹	**Resource.AssignVMToPool** **VApp.Import** **VirtualMachine.Config.AddExistingDisk** **VirtualMachine.Config.AddNewDisk** **VirtualMachine.Config.AddRemoveDevice** **VirtualMachine.Config.AdvancedConfig** **VirtualMachine.Config.Annotation** **VirtualMachine.Config.CPUCount** **VirtualMachine.Config.DiskExtend** **VirtualMachine.Config.DiskLease** **VirtualMachine.Config.EditDevice** **VirtualMachine.Config.Memory** **VirtualMachine.Config.RemoveDisk** **VirtualMachine.Config.Rename** **VirtualMachine.Config.ResetGuestInfo** **VirtualMachine.Config.Resource** **VirtualMachine.Config.Settings** **VirtualMachine.Config.UpgradeVirtualHardware** **VirtualMachine.Interact.GuestControl** **VirtualMachine.Interact.PowerOff** **VirtualMachine.Interact.PowerOn** **VirtualMachine.Interact.Reset** **VirtualMachine.Inventory.Create** **VirtualMachine.Inventory.CreateFromExisting** **VirtualMachine.Inventory.Delete** **VirtualMachine.Provisioning.Clone** **Folder.Create** **Folder.Delete**

此外，用户需要一些 **ReadOnly** 权限，某些角色需要权限来提升对子对象的权限。这些设置会根据您是否将集群安装到现有文件夹而有所不同。

例 11.12. 所需的权限和传播设置

vSphere 对象	文件夹类型	传播到子对象	所需的权限
vSphere vCenter	Always	False	列出所需的权限
vSphere vCenter Datacenter	现有文件夹	False	**ReadOnly** 权限
	安装程序创建文件夹	True	列出所需的权限
vSphere vCenter Cluster	Always	True	列出所需的权限
vSphere vCenter Datastore	Always	False	列出所需的权限
vSphere Switch	Always	False	**ReadOnly** 权限
vSphere 端口组	Always	False	列出所需的权限
vSphere vCenter Virtual Machine Folder	现有文件夹	True	列出所需的权限

有关只使用所需权限创建帐户的更多信息，请参阅 vSphere 文档中的 vSphere 权限和用户管理任务。

将 OpenShift Container Platform 与 vMotion 搭配使用

如果要在 vSphere 环境中使用 vMotion，请在安装 OpenShift Container Platform 集群前考虑以下内容。

- OpenShift Container Platform 通常支持仅用于计算的 vMotion。使用 Storage vMotion 可能会导致问题且不被支持。
  为了帮助确保计算和 control plane 节点的正常运行时间，建议您遵循 VMware 最佳实践进行 vMotion。还建议使用 VMware 反关联性规则来改进 OpenShift Container Platform 在维护或硬件问题期间的可用性。

  有关 vMotion 和 anti-affinity 规则的更多信息，请参阅 VMware vSphere 文档 了解 vMotion 网络要求和虚拟机反关联性规则。

- 如果您在 pod 中使用 vSphere 卷，请手动或通过 Storage vMotion 在数据存储间迁移虚拟机，从而导致 OpenShift Container Platform 持久性卷(PV)对象中的无效引用。这些引用可防止受影响的 pod 启动，并可能导致数据丢失。

- 同样，OpenShift Container Platform 不支持在数据存储间有选择地迁移 VMDK、使用数据存储集群进行虚拟机置备、动态或静态置备 PV，或使用作为数据存储集群一部分的数据存储进行 PV 的动态或静态置备。

集群资源

当部署使用安装程序置备的基础架构的 OpenShift Container Platform 集群时，安装程序必须能够在 vCenter 实例中创建多个资源。

标准 OpenShift Container Platform 安装会创建以下 vCenter 资源：

- 1 个文件夹
- 1 标签（Tag）类别
- 1 个标签（Tag）
- 虚拟机:
  - 1 个模板
  - 1 个临时 bootstrap 节点
  - 3 个 control plane 节点
  - 3 个计算机器

虽然这些资源使用了 856 GB 存储，但 bootstrap 节点会在集群安装过程中被销毁。使用标准集群至少需要 800 GB 存储。

如果部署了更多计算机器，OpenShift Container Platform 集群将使用更多存储。

集群的限制
可用资源因集群而异。vCenter 中可能的集群数量主要受可用存储空间以及对所需资源数量的限制。确保考虑集群创建的 vCenter 资源的限制和部署集群所需的资源，如 IP 地址和网络。

网络要求
网络必须使用 DHCP，并确保 DHCP 服务器被配置为为集群机器提供持久的 IP 地址。

注意

在安装开始前，持久性 IP 地址不可用。分配 DHCP 范围后，在安装后使用持久 IP 地址手动替换分配。

受限网络中的虚拟机必须有权访问 vCenter，以便它可以置备和管理节点、持久性卷声明（PVC）和其他资源。另外，在安装 OpenShift Container Platform 集群前，必须创建以下网络资源：

注意

建议集群中的每个 OpenShift Container Platform 节点都可以访问可通过 DHCP 发现的网络时间协议（NTP）服务器。没有 NTP 服务器也可安装。但是，异步服务器时钟将导致错误，NTP 服务器会阻止。

所需的 IP 地址
安装程序置备的 vSphere 安装需要这些静态 IP 地址：

- API 地址用于访问集群 API。
- Ingress 地址用于集群入口流量。
- 当将集群从版本 4.5 升级到 4.6 时，会使用 control plane 节点地址。

安装 OpenShift Container Platform 集群时，必须向安装程序提供这些 IP 地址。

DNS 记录

您必须在正确的 DNS 服务器中为托管 OpenShift Container Platform 集群的 vCenter 实例创建两个静态 IP 地址的 DNS 记录。在每个记录中，**<cluster_name>** 是集群名称，**<base_domain>** 是您在安装集群时指定的集群基域。完整的 DNS 记录采用如下格式: **<component>.<cluster_name>.<base_domain>.**。

表 11.56. 所需的 DNS 记录

组件	记录	描述
API VIP	**api.<cluster_name>.<base_domain>.**	此 DNS A/AAAA 或 CNAME 记录必须指向 control plane 机器的负载均衡器。此记录必须能由集群外的客户端和集群内的所有节点解析。
Ingress VIP	***.apps.<cluster_name>.<base_domain>.**	通配符 DNS A/AAAA 或 CNAME 记录，指向以运行入口路由器 Pod 的机器（默认为 worker 节点）为目标的负载均衡器。此记录必须能由集群外的客户端和集群内的所有节点解析。

11.6.7. 生成 SSH 私钥并将其添加到代理中

如果要在集群上执行安装调试或灾难恢复，则必须为 **ssh-agent** 和安装程序提供 SSH 密钥。您可以使用此密钥访问公共集群中的 bootstrap 机器来排除安装问题。

注意

在生产环境中，您需要进行灾难恢复和调试。

您可以使用此密钥以 **core** 用户身份通过 SSH 连接到 master 节点。在部署集群时，此密钥会添加到 **core** 用户的 **~/.ssh/authorized_keys** 列表中。

注意

您必须使用一个本地密钥，而不要使用在特定平台上配置的密钥，如 AWS 密钥对。

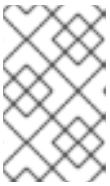
流程

1. 如果还没有为计算机上免密码身份验证而配置的 SSH 密钥，请创建一个。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ ssh-keygen -t ed25519 -N '' \
  -f <path>/<file_name> 1
```

1 指定新 SSH 密钥的路径和文件名，如 **~/.ssh/id_rsa**。如果您已有密钥对，请确保您的公钥位于 **~/.ssh** 目录中。

运行此命令会在指定的位置生成不需要密码的 SSH 密钥。



注意

如果您计划在 **x86_64** 架构中安装使用 FIPS 验证的/Modules in Process 加密库的 OpenShift Container Platform 集群，不要创建使用 **ed25519** 算法的密钥。反之，创建一个使用 **rsa** 或 **ecdsa** 算法的密钥。

2. 作为后台任务启动 **ssh-agent** 进程：

```
$ eval "$(ssh-agent -s)"
```

输出示例

```
Agent pid 31874
```

注意

如果您的集群采用 FIPS 模式，则只使用 FIPS 兼容算法来生成 SSH 密钥。密钥必须是 RSA 或 ECDSA。

3. 将 SSH 私钥添加到 **ssh-agent**：

```
$ ssh-add <path>/<file_name> 1
```

输出示例

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

1 指定 SSH 私钥的路径和文件名，如 **~/.ssh/id_rsa**

后续步骤

- 在安装 OpenShift Container Platform 时，为安装程序提供 SSH 公钥。

11.6.8. 在您的系统信任中添加 vCenter root CA 证书

由于安装程序需要访问 vCenter 的 API，所以必须在安装 OpenShift Container Platform 集群前将 vCenter 的可信 root CA 证书添加到系统信任中。

流程

1. 在 vCenter 主页中下载 vCenter 的 root CA 证书。在 vSphere Web Services SDK 部分点击 **Download trusted root CA certificates**。**<vCenter>/certs/download.zip** 文件下载。

2. 提取包含 vCenter root CA 证书的压缩文件。压缩文件的内容类似以下文件结构：

```
certs
├── lin
│   ├── 108f4d17.0
```

```
│   ├── 108f4d17.r1
│   ├── 7e757f6a.0
│   ├── 8e4f8471.0
│   └── 8e4f8471.r0
├── mac
│   ├── 108f4d17.0
│   ├── 108f4d17.r1
│   ├── 7e757f6a.0
│   ├── 8e4f8471.0
│   └── 8e4f8471.r0
└── win
    ├── 108f4d17.0.crt
    ├── 108f4d17.r1.crl
    ├── 7e757f6a.0.crt
    ├── 8e4f8471.0.crt
    └── 8e4f8471.r0.crl

3 directories, 15 files
```

3. 将您的操作系统的文件添加到系统信任中。例如，在 Fedora 操作系统上运行以下命令：

```
# cp certs/lin/* /etc/pki/ca-trust/source/anchors
```

4. 更新您的系统信任关系。例如，在 Fedora 操作系统上运行以下命令：

```
# update-ca-trust extract
```

11.6.9. 为受限网络安装创建 RHCOS 镜像

下载 Red Hat Enterprise Linux CoreOS（RHCOS）镜像，以便在受限网络 VMware vSphere 环境中安装 OpenShift Container Platform。

先决条件

- 获取 OpenShift Container Platform 安装程序。对于受限网络安装，该程序位于您的镜像 registry 主机上。

流程

1. 登录到红帽客户门户网站的产品下载页。
2. 在 **Version** 下，为 RHEL 8 选择 OpenShift Container Platform 4.6 的最新发行版本。

重要

RHCOS 镜像可能不会随着 OpenShift Container Platform 的每一发行版本都有改变。您必须下载最高版本的镜像，其版本号应小于或等于您安装的 OpenShift Container Platform 版本。如果可用，请使用与 OpenShift Container Platform 版本匹配的镜像版本。

3. 下载 **Red Hat Enterprise Linux CoreOS(RHCOS)- vSphere**镜像。
4. 将您下载的镜像上传到堡垒服务器可访问的位置。

该镜像现在可用于受限安装。记录 OpenShift Container Platform 部署中使用的镜像名称或位置。

11.6.10. 创建安装配置文件

您可以自定义在 VMware vSphere 上安装的 OpenShift Container Platform 集群。

先决条件

- 获取 OpenShift Container Platform 安装程序以及集群的 pull secret。对于受限网络安装，这些文件位于您的堡垒主机上。
- 具有创建镜像容器镜像仓库（registry）时生成的 **imageContentSources** 值。
- 获取您的镜像 registry 的证书内容。
- 检索 Red Hat Enterprise Linux CoreOS（RHCOS）镜像，并将其上传到可访问的位置。

流程

1. 创建 **install-config.yaml** 文件。
   a. 更改到包含安装程序的目录，再运行以下命令：

```
$ ./openshift-install create install-config --dir <installation_directory> 1
```

   1 对于 **<installation_directory>**，请指定用于保存安装程序所创建的文件的目录名称。

重要

指定一个空目录。一些安装信息，如 bootstrap X.509 证书，有较短的过期间隔，因此不要重复使用安装目录。如果要重复使用另一个集群安装中的个别文件，可以将其复制到您的目录中。但是，一些安装数据的文件名可能会在发行版本之间有所改变。从 OpenShift Container Platform 老版本中复制安装文件时要格外小心。

   b. 在提示符处，提供您的云的配置详情：
      i. 可选：选择用来访问集群机器的 SSH 密钥。

注意

对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 **ssh-agent** 进程使用的 SSH 密钥。

      ii. 选择 **vsphere** 作为目标平台。
      iii. 指定 vCenter 实例的名称。
      iv. 指定创建集群所需的权限的 vCenter 帐户的用户名和密码。
      安装程序连接到您的 vCenter 实例。
      v. 选择要连接的 vCenter 实例中的数据中心。

vi. 选择要使用的默认 vCenter 数据存储。

vii. 选择要在其中安装 vCenter 集群的 OpenShift Container Platform 集群。安装程序使用 vSphere 集群的 root 资源池作为默认资源池。

viii. 选择包含您配置的虚拟 IP 地址和 DNS 记录的 vCenter 实例中的网络。

ix. 输入您为 control plane API 访问配置的虚拟 IP 地址。

x. 输入您为集群入口配置的虚拟 IP 地址。

xi. 输入基域。这个基域必须与您配置的 DNS 记录中使用的域相同。

xii. 为集群输入一个描述性名称。集群名称必须与您配置的 DNS 记录中使用的相同。

xiii. 粘贴 Red Hat OpenShift Cluster Manager 中的 pull secret 。

2. 在 **install-config.yaml** 文件中，将 **platform.vsphere.clusterOSImage** 的值设置为镜像位置或名称。例如：

```
platform:
  vsphere:
    clusterOSImage: http://mirror.example.com/images/rhcos-43.81.201912131630.0-
vmware.x86_64.ova?
sha256=ffebbd68e8a1f2a245ca19522c16c86f67f9ac8e4e0c1f0a812b068b16f7265d
```

3. 编辑 **install-config.yaml** 文件，以提供在受限网络中安装所需的其他信息。

a. 更新 **pullSecret** 值，使其包含 registry 的身份验证信息：

```
pullSecret: '{"auths":{"<mirror_host_name>:5000": {"auth": "<credentials>","email":
"you@example.com"}}}'
```

对于 **<mirror_host_name>**，请指定您在镜像 registry 证书中指定的 registry 域名；对于 **<credentials>**，请指定您的镜像 registry 的 base64 编码用户名和密码。

b. 添加 **additionalTrustBundle** 参数和值。

```
additionalTrustBundle: |
  -----BEGIN CERTIFICATE-----

ZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZ
  -----END CERTIFICATE-----
```

该值必须是您用于镜像 registry 的证书文件内容，可以是现有的可信证书颁发机构或您为镜像 registry 生成的自签名证书。

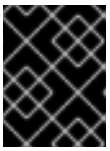
c. 添加镜像内容资源，如下例所示：

```
imageContentSources:
- mirrors:
  - <mirror_host_name>:5000/<repo_name>/release
  source: quay.example.com/openshift-release-dev/ocp-release
- mirrors:
  - <mirror_host_name>:5000/<repo_name>/release
  source: registry.example.com/ocp/release
```

要完成这些值，请使用您在创建镜像容器镜像仓库（registry）时记录的 **imageContentSources**。

4. 对需要的 **install-config.yaml** 文件做任何其他修改。您可以在**安装配置参数**部分中找到有关可用参数的更多信息。

5. 备份 **install-config.yaml** 文件，以便用于安装多个集群。



重要

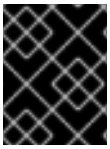
install-config.yaml 文件会在安装过程中消耗掉。如果要重复使用此文件，必须现在备份。

11.6.10.1. 安装配置参数

在部署 OpenShift Container Platform 集群前，您可以提供参数值，以描述托管集群的云平台的帐户并选择性地自定义集群平台。在创建 **install-config.yaml** 安装配置文件时，您可以通过命令行来提供所需的参数的值。如果要自定义集群，可以修改 **install-config.yaml** 文件来提供关于平台的更多信息。

注意

安装之后，您无法修改 **install-config.yaml** 文件中的这些参数。



重要

openshift-install 命令不验证参数的字段名称。如果指定了不正确的名称，则不会创建相关的文件或对象，且不会报告错误。确保所有指定的参数的字段名称都正确。

11.6.10.1.1. 所需的配置参数

下表描述了所需的安装配置参数：

表 11.57. 所需的参数

参数	描述	值
apiVersion	**install-config.yaml** 内容的 API 版本。当前版本是 **v1**。安装程序还可能支持旧的 API 版本。	字符串
baseDomain	云供应商的基域。此基础域用于创建到 OpenShift Container Platform 集群组件的路由。集群的完整 DNS 名称是 **baseDomain** 和 **metadata.name** 参数值的组合，其格式为 **<metadata.name>.<baseDomain>**。	完全限定域名或子域名，如 **example.com**。

参数	描述	值
metadata	Kubernetes 资源 **ObjectMeta**，其中只消耗 **name** 参数。	对象
metadata.name	集群的名称。集群的 DNS 记录是 **{{.metadata.name}}.{{.baseDomain}}** 的子域。	小写字母和连字符(**-**)的字符串，如 **dev**。
platform	执行安装的具体平台配置：**aws**、**baremetal**、**azure**、**openstack**、**ovirt**、**vsphere**。有关 **platform.<platform>** 参数的额外信息，请参考下表来了解您的具体平台。	对象

pullSecret	从 Red Hat OpenShift Cluster Manager 获取 pull secret，验证从 Quay.io 等服务中下载 OpenShift Container Platform 组件的容器镜像。	<pre>{
		  "auths":{
		  "cloud.openshift.com":{
		  "auth":"b3Blb=",
		  "email":"you@example.com"
		  },
		  "quay.io":{
		  "auth":"b3Blb=",
		  "email":"you@example.com"
		  }
		  }
		}</pre>

11.6.10.1.2. 网络配置参数

您可以根据现有网络基础架构的要求自定义安装配置。例如，您可以扩展集群网络的 IP 地址块，或者提供不同于默认值的不同 IP 地址块。

只支持 IPv4 地址。

表 11.58. 网络参数

参数	描述	值
networking	集群网络的配置。	对象 **注意** 您不能在安装后修改 **networking** 对象指定的参数。

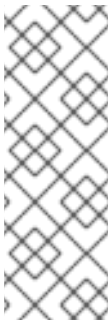
参数	描述	值
networking.networkType	要安装的集群网络供应商 Container Network Interface（CNI）插件。	**OpenShiftSDN** 或 **OVNKubernetes**。默认值为 **OpenShiftSDN**。
networking.clusterNetwork	pod 的 IP 地址块。 默认值为 **10.128.0.0/14**，主机前缀为 **/23**。 如果您指定多个 IP 地址块，则块不得互相重叠。	一个对象数组。例如： networking: clusterNetwork: - cidr: 10.128.0.0/14 hostPrefix: 23
networking.clusterNetwork.cidr	使用 **networking.clusterNetwork** 时需要此项。IP 地址块。 一个 IPv4 网络。	使用 CIDR 形式的 IP 地址块。IPv4 块的前缀长度介于 **0** 到 **32** 之间。
networking.clusterNetwork.hostPrefix	分配给每个单独节点的子网前缀长度。例如，如果 **hostPrefix** 设为 **23**，则每个节点从所给的 **cidr** 中分配一个 **/23** 子网。**hostPrefix** 值 **23** 提供 510（2^(32 - 23)- 2）个 pod IP 地址。	子网前缀。 默认值为 **23**。
networking.serviceNetwork	服务的 IP 地址块。默认值为 **172.30.0.0/16**。 OpenShift SDN 和 OVN-Kubernetes 网络供应商只支持服务网络的一个 IP 地址块。	CIDR 格式具有 IP 地址块的数组。例如： networking: serviceNetwork: - 172.30.0.0/16
networking.machineNetwork	机器的 IP 地址块。 如果您指定多个 IP 地址块，则块不得互相重叠。	一个对象数组。例如： networking: machineNetwork: - cidr: 10.0.0.0/16
networking.machineNetwork.cidr	使用 **networking.machineNetwork** 时需要。IP 地址块。libvirt 以外的所有平台的默认值为 **10.0.0.0/16**。对于 libvirt，默认值为 **192.168.126.0/24**。	CIDR 表示法中的 IP 网络块。 例如：**10.0.0.0/16**。 **注意** 将 **networking.machineNetwork** 设置为与首选 NIC 所在的 CIDR 匹配。

11.6.10.1.3. 可选配置参数

下表描述了可选安装配置参数：

表 11.59. 可选参数

参数	描述	值
additionalTrustBundle	添加到节点可信证书存储中的 PEM 编码 X.509 证书捆绑包。配置了代理时，也可以使用这个信任捆绑包。	字符串
compute	组成计算节点的机器的配置。	machine-pool 对象的数组。详情请查看以下"Machine-pool"表。
compute.architecture	决定池中机器的指令集合架构。目前不支持异构集群，因此所有池都必须指定相同的架构。有效值为 **amd64** （默认值）。	字符串
compute.hyperthreading	是否在计算机器上启用或禁用并发多线程或**超线程**。默认情况下，启用并发多线程以提高机器内核的性能。  **重要** 如果禁用并发多线程，请确保在容量规划时考虑到机器性能可能会显著降低的问题。	**Enabled** 或 **Disabled**
compute.name	使用 **compute** 时需要此值。机器池的名称。	**worker**
compute.platform	使用 **compute** 时需要此值。使用此参数指定托管 worker 机器的云供应商。此参数值必须与 **controlPlane.platform** 参数值匹配。	**aws**、**azure**、**gcp**、**openstack**、**ovirt**、**vsphere** 或 **{}**
compute.replicas	要置备的计算机器数量，也称为 worker 机器。	大于或等于 **2** 的正整数。默认值为 **3**。
controlPlane	组成 control plane 的机器的配置。	**MachinePool** 对象的数组。详情请查看以下"Machine-pool"表。
controlPlane.architecture	决定池中机器的指令集合架构。目前不支持异构集群，因此所有池都必须指定相同的架构。有效值为 **amd64** （默认值）。	字符串

参数	描述	值
controlPlane.hyperthreading	是否在 control plane 机器上启用或禁用并发多线程或**超线程**。默认情况下，启用并发多线程以提高机器内核的性能。  **重要** 如果禁用并发多线程，请确保在容量规划时考虑到机器性能可能会显著降低的问题。	**Enabled** 或 **Disabled**
controlPlane.name	使用 **controlPlane** 时需要。机器池的名称。	**master**
controlPlane.platform	使用 **controlPlane** 时需要。使用此参数指定托管 control plane 机器的云供应商。此参数值必须与 **compute.platform** 参数值匹配。	**aws**、**azure**、**gcp**、**openstack**、**ovirt**、**vsphere** 或 **{}**
controlPlane.replicas	要置备的 control plane 机器数量。	唯一支持的值是 **3**，它是默认值。
credentialsMode	Cloud Credential Operator（CCO）模式。如果没有指定任何模式，CCO 会动态地尝试决定提供的凭证的功能，在支持多个模式的平台上使用 mint 模式。  **注意** 不是所有 CCO 模式都支持所有云供应商。如需有关 CCO 模式的更多信息，请参阅*Red Hat Operator 参考指南*内容中的 *Cloud Credential Operator* 条目。	**Mint**、**Passthrough**、**Manual** 或空字符串(**""**)。

参数	描述	值
fips	启用或禁用 FIPS 模式。默认为 **false**（禁用）。如果启用了 FIPS 模式，运行 OpenShift Container Platform 的 Red Hat Enterprise Linux CoreOS (RHCOS) 机器会绕过默认的 Kubernetes 加密套件，并使用由 RHCOS 提供的加密模块。  **重要** 只有在 **x86_64** 架构中的 OpenShift Container Platform 部署支持 FIPS 验证的/Modules in Process 加密库。  **注意** 如果使用 Azure File 存储，则无法启用 FIPS 模式。	**false** 或 **true**
imageContentSources	release-image 内容的源和仓库。	对象数组。包括一个 **source** 以及可选的 **mirrors**，如下表所示。
imageContentSources.source	使用 **imageContentSources** 时需要。指定用户在镜像拉取规格中引用的仓库。	字符串
imageContentSources.mirrors	指定可能还包含同一镜像的一个或多个仓库。	字符串数组
publish	如何发布或公开集群的面向用户的端点，如 Kubernetes API、OpenShift 路由。	**Internal** 或 **External**。默认值为 **External**。 在非云平台上不支持将此字段设置为 **Internal**。  **重要** 如果将字段的值设为 **Internal**，集群将无法运行。如需更多信息，请参阅 BZ#1953035。

参数	描述	值
sshKey	用于验证集群机器访问的 SSH 密钥或密钥。  **注意** 对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 **ssh-agent** 进程使用的 SSH 密钥。	一个或多个密钥。例如： sshKey: <key1> <key2> <key3>

11.6.10.1.4. 其他 VMware vSphere 配置参数

下表描述了其他 VMware vSphere 配置参数：

表 11.60. 其他 VMware vSphere 集群参数

参数	描述	值
platform.vsphere.vCenter	vCenter 服务器的完全限定主机名或 IP 地址。	字符串
platform.vsphere.username	用于连接 vCenter 实例的用户名。此用户必须至少具有 vSphere 中 静态或动态持久性卷置备 所需的角色和权限。	字符串
platform.vsphere.password	vCenter 用户名的密码。	字符串
platform.vsphere.datacenter	要在 vCenter 实例中使用的数据中心的名称。	字符串
platform.vsphere.defaultDatastore	用于置备卷的默认数据存储名称。	字符串
platform.vsphere.folder	*可选*。安装程序创建虚拟机的现有文件夹的绝对路径。如果没有提供这个值，安装程序会创建一个文件夹，它的名称是数据中心虚拟机文件夹中的基础架构 ID。	字符串，如 **/<datacenter_name>/vm/<folder_name>/<subfolder_name>**。
platform.vsphere.network	包含您配置的虚拟 IP 地址和 DNS 记录的 vCenter 实例中的网络。	字符串
platform.vsphere.cluster	在其中安装 OpenShift Container Platform 集群的 vCenter 集群。	字符串

参数	描述	值
platform.vsphere.apiVIP	为 control plane API 访问配置的虚拟 IP（VIP）地址。	IP 地址，如 **128.0.0.1**。
platform.vsphere.ingressVIP	为集群入口配置的虚拟 IP（VIP）地址。	IP 地址，如 **128.0.0.1**。

11.6.10.1.5. 可选的 VMware vSphere 机器池配置参数

下表描述了可选的 VMware vSphere 机器池配置参数：

表 11.61. 可选的 VMware vSphere 机器池参数

参数	描述	值
platform.vsphere.clusterOSImage	安装程序从中下载 RHCOS 镜像的位置。您必须设置此参数以便在受限网络中执行安装。	HTTP 或 HTTPS URL，可选使用 SHA-256 checksum。例如：**https://mirror.openshift.com/images/rhcos-<version>-vmware.<architecture>.ova**。
platform.vsphere.osDisk.diskSizeGB	以 GB 为单位的磁盘大小。	整数
platform.vsphere.cpus	分配虚拟机的虚拟处理器内核总数。	整数
platform.vsphere.coresPerSocket	虚拟机中每个插槽的内核数。虚拟机上的虚拟套接字数量为 **platform.vsphere.cpus**/**platform.vsphere.coresPerSocket**。默认值为 **1**。	整数
platform.vsphere.memoryMB	以 MB 为单位的虚拟机内存大小。	整数

11.6.10.2. 安装程序置备的 VMware vSphere 集群的 install-config.yaml 文件示例

您可以自定义 install-config.yaml 文件，以指定有关 OpenShift Container Platform 集群平台的更多信息，或修改所需参数的值。

```
apiVersion: v1
baseDomain: example.com 1
compute: 2
- hyperthreading: Enabled 3
  name: worker
  replicas: 3
  platform:
    vsphere: 4
```

```
      cpus: 2
      coresPerSocket: 2
      memoryMB: 8192
      osDisk:
        diskSizeGB: 120
controlPlane: 5
  hyperthreading: Enabled 6
  name: master
  replicas: 3
  platform:
    vsphere: 7
      cpus: 4
      coresPerSocket: 2
      memoryMB: 16384
      osDisk:
        diskSizeGB: 120
metadata:
  name: cluster 8
platform:
  vsphere:
    vcenter: your.vcenter.server
    username: username
    password: password
    datacenter: datacenter
    defaultDatastore: datastore
    folder: folder
    network: VM_Network
    cluster: vsphere_cluster_name 9
    apiVIP: api_vip
    ingressVIP: ingress_vip
    clusterOSImage: http://mirror.example.com/images/rhcos-48.83.202103221318-0-
vmware.x86_64.ova 10
fips: false
pullSecret: '{"auths":{"<local_registry>": {"auth": "<credentials>","email": "you@example.com"}}}' 11
sshKey: 'ssh-ed25519 AAAA...'
additionalTrustBundle: | 12
  -----BEGIN CERTIFICATE-----
  ZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZ
  -----END CERTIFICATE-----
imageContentSources: 13
- mirrors:
  - <local_registry>/<local_repository_name>/release
  source: quay.io/openshift-release-dev/ocp-release
- mirrors:
  - <local_registry>/<local_repository_name>/release
  source: quay.io/openshift-release-dev/ocp-v4.0-art-dev
```

1 集群的基域。所有 DNS 记录都必须是这个基域的子域，并包含集群名称。

2 5 **controlPlane** 部分是一个单映射，但 **compute** 部分是一系列映射。为满足不同数据结构的要求，**compute** 部分的第一行必须以连字符 **-** 开头，**controlPlane** 部分的第一行则不可以连字符开头。只使用一个 control plane 池。

3 6

是否要启用或禁用并发多线程或超线程。默认情况下，启用并发多线程以提高机器内核的性能。您可以通过将参数值设为 **Disabled** 来禁用。如果您在某些集群机器上禁用并发多线程，则必须在所有集群机器上禁用。

重要

如果禁用并发多线程，请确保在容量规划时考虑到机器性能可能会显著降低的问题。如果您禁用并发多线程，则计算机必须至少使用 8 个 CPU 和 32GB RAM。

4 7 可选：为 compute 和 control plane 机器提供额外的机器池参数配置。

8 您在 DNS 记录中指定的集群名称。

9 要在其中安装 OpenShift Container Platform 集群的 vSphere 集群。安装程序使用 vSphere 集群的 root 资源池作为默认资源池。

10 堡垒服务器可以访问的 Red Hat Enterprise Linux CoreOS（RHCOS）镜像的位置。

11 对于 **<local_registry>**，请指定 registry 域名，以及您的镜像 registry 用来提供内容的可选端口。例如： **registry.example.com** 或者 **registry.example.com:5000**。使用 **<credentials>** 为您生成的镜像 registry 指定 base64 编码的用户名和密码。

12 提供用于镜像 registry 的证书文件内容。

13 提供命令输出中的 **imageContentSources** 部分来镜像存储库。

11.6.10.3. 在安装过程中配置集群范围代理

生产环境可能会拒绝直接访问互联网，而是提供 HTTP 或 HTTPS 代理。您可以通过在 **install-config.yaml** 文件中配置代理设置，将新的 OpenShift Container Platform 集群配置为使用代理。

先决条件

- 您有一个现有的 **install-config.yaml** 文件。
- 您检查了集群需要访问的站点，并决定是否需要绕过代理。默认情况下代理所有集群出口流量，包括对托管云供应商 API 的调用。您需要将站点添加到 **Proxy** 对象的 **spec.noProxy** 字段来绕过代理。

注意

Proxy 对象 **status.noProxy** 字段使用安装配置中的 **networking.machineNetwork[].cidr**、**networking.clusterNetwork[].cidr** 和 **networking.serviceNetwork[]** 字段的值填充。

对于在 Amazon Web Services(AWS)、Google Cloud Platform(GCP)、Microsoft Azure 和 Red Hat OpenStack Platform(RHOSP)上安装, **Proxy** 对象 **status.noProxy** 字段也会使用实例元数据端点填充(**169.254.169.254**)。

流程

1. 编辑 **install-config.yaml** 文件并添加代理设置。例如：

```
apiVersion: v1
```

```
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: example.com 3
additionalTrustBundle: | 4
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
...
```

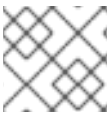
1. 用于创建集群外 HTTP 连接的代理 URL。URL 必须是 **http**。
2. 用于创建集群外 HTTPS 连接的代理 URL。
3. 要排除在代理中的目标域名、IP 地址或其他网络 CIDR 的逗号分隔列表。在域前面加 **.** 来仅匹配子域。例如： **.y.com** 匹配 **x.y.com**，但不匹配 **y.com**。使用 * 绕过所有目的地的代理。您必须包含 vCenter 的 IP 地址以及用于其机器的 IP 范围。
4. 如果提供，安装程序会在 **openshift-config 命名空间中生成名为 user-ca- bundle** 的配置映射来保存额外的 CA 证书。如果您提供 **additionalTrustBundle** 和至少一个代理设置，则 **Proxy** 对象会被配置为引用 **trustedCA** 字段中的 **user-ca-bundle** 配置映射。然后，Cluster Network Operator 会创建一个 **trusted-ca-bundle** 配置映射，该配置映射将为 **trustedCA** 参数指定的内容与 RHCOS 信任捆绑包合并。 **additionalTrustBundle** 字段是必需的，除非代理的身份证书由来自 RHCOS 信任捆绑包的颁发机构签名。

注意

安装程序不支持代理的 **readinessEndpoints** 字段。

2. 保存该文件，并在安装 OpenShift Container Platform 时引用。

安装程序会创建一个名为 **cluster** 的集群范围代理，该代理使用提供的 **install-config.yaml** 文件中的代理设置。如果没有提供代理设置，仍然会创建一个 **cluster Proxy** 对象，但它会有一个空 **spec**。



注意

只支持名为 **cluster** 的 **Proxy** 对象，且无法创建额外的代理。

11.6.11. 部署集群

您可以在兼容云平台中安装 OpenShift Container Platform。

重要

安装程序的 **create cluster** 命令只能在初始安装过程中运行一次。

先决条件

- 获取 OpenShift Container Platform 安装程序以及集群的 pull secret。

流程

1. 更改为包含安装程序的目录并初始化集群部署：

```
$ ./openshift-install create cluster --dir <installation_directory> \ 1
    --log-level=info 2
```

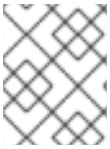
1 对于 **<installation_directory>**，请指定自定义 **./install-config.yaml** 文件的位置。

2 要查看不同的安装详情，请指定 **warn**、**debug** 或 **error**，而不要指定 **info**。

集群部署完成后，终端会显示访问集群的信息，包括指向其 Web 控制台的链接和 **kubeadmin** 用户的凭证。

```
...
INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run 'export
KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'
INFO Access the OpenShift web-console here: https://console-openshift-
console.apps.mycluster.example.com
INFO Login to the console with user: "kubeadmin", and password: "4vYBz-Ee6gm-ymBZj-Wt5AL"
INFO Time elapsed: 36m22s
```

+



注意

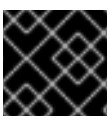
当安装成功时，集群访问和凭证信息还会输出到 **<installation_directory>/.openshift_install.log**。

+

重要

- 安装程序生成的 Ignition 配置文件包含在 24 小时后过期的证书，然后在过期时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外情况是，您需要手动批准待处理的 **node-bootstrapper** 证书签名请求（CSR）来恢复 kubelet 证书。如需更多信息，请参阅*从过期的 control plane 证书中恢复*的文档。
- 建议您在生成 12 小时后使用 Ignition 配置文件，因为集群安装后 24 小时证书从 16 小时轮转至 22 小时。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中运行证书更新时避免安装失败。

+



重要

您不得删除安装程序或安装程序所创建的文件。需要这两者才能删除集群。

11.6.12. 通过下载二进制文件安装 OpenShift CLI

您需要安装 CLI（**oc**）来使用命令行界面与 OpenShift Container Platform 进行交互。您可在 Linux、Windows 或 macOS 上安装 **oc**。

重要

如果安装了旧版本的 **oc**，则无法使用 OpenShift Container Platform 4.6 中的所有命令。下载并安装新版本的 **oc**。

11.6.12.1. 在 Linux 上安装 OpenShift CLI

您可以按照以下流程在 Linux 上安装 OpenShift CLI（**oc**）二进制文件。

流程

1. 进入到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 在 **Version** 下拉菜单中选择相应的版本。
3. 单击 **OpenShift v4.6 Linux 客户端** 条目旁边的 **Download Now**，再保存文件。
4. 解包存档：

```
$ tar xvzf <file>
```

5. 把 **oc** 二进制代码放到 **PATH** 中的目录中。
   执行以下命令可以查看当前的 **PATH** 设置：

```
$ echo $PATH
```

安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
$ oc <command>
```

11.6.12.2. 在 Windows 上安装 OpenShift CLI

您可以按照以下流程在 Windows 上安装 OpenShift CLI（**oc**）二进制代码。

流程

1. 进入到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 在 **Version** 下拉菜单中选择相应的版本。
3. 单击 **OpenShift v4.6 Windows 客户端**条目旁边的 **Download Now**，再保存文件。
4. 使用 ZIP 程序解压存档。
5. 把 **oc** 二进制代码放到 **PATH** 中的目录中。
   要查看您的 **PATH**，请打开命令提示窗口并执行以下命令：

```
C:\> path
```

安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
C:\> oc <command>
```

11.6.12.3. 在 macOS 上安装 OpenShift CLI

您可以按照以下流程在 macOS 上安装 OpenShift CLI（**oc**）二进制代码。

流程

1. 进入到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 在 **Version** 下拉菜单中选择相应的版本。
3. 单击 **OpenShift v4.6 MacOSX 客户端**条目旁边的 **Download Now**，再保存文件。
4. 解包和解压存档。
5. 将 **oc** 二进制文件移到 PATH 的目录中。
   要查看您的 **PATH**，打开一个终端窗口并执行以下命令：

```
$ echo $PATH
```

安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
$ oc <command>
```

11.6.13. 使用 CLI 登录到集群

您可以通过导出集群 **kubeconfig** 文件，以默认系统用户身份登录集群。**kubeconfig** 文件包含关于集群的信息，供 CLI 用于将客户端连接到正确集群和 API 服务器。该文件特只适用于一个特定的集群，在 OpenShift Container Platform 安装过程中创建。

先决条件

- 已部署了 OpenShift Container Platform 集群。
- 已安装 **oc** CLI。

流程

1. 导出 **kubeadmin** 凭证：

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

2. 使用导出的配置，验证能否成功运行 **oc** 命令：

```
$ oc whoami
```

输出示例

```
system:admin
```

11.6.14. 禁用默认的 OperatorHub 源

在 OpenShift Container Platform 安装过程中，默认为 OperatorHub 配置由红帽和社区项目提供的源内容的 operator 目录。在受限网络环境中，必须以集群管理员身份禁用默认目录。

流程

- 通过在 **OperatorHub** 对象中添加 **disableAllDefaultSources: true** 来禁用默认目录的源：

```
$ oc patch OperatorHub cluster --type json \
    -p '[{"op": "add", "path": "/spec/disableAllDefaultSources", "value": true}]'
```

提示

或者，您可以使用 Web 控制台管理目录源。在 **Administration → Cluster Settings → Global Configuration → OperatorHub** 页面中，点 **Sources** 选项卡，其中可创建、删除、禁用和启用单独的源。

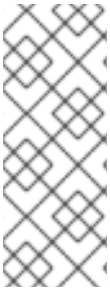
11.6.15. 创建 registry 存储

安装集群后，必须为 Registry Operator 创建存储。

11.6.15.1. 安装过程中删除的镜像 registry

在不提供可共享对象存储的平台上，OpenShift Image Registry Operator bootstraps 本身的状态是 **Removed**。这允许 **openshift-installer** 在这些平台类型上完成安装。

将 **ManagementState** Image Registry Operator 配置从 **Removed** 改为 **Managed**。



注意

Prometheus 控制台提供了一个 **ImageRegistryRemoved** 警报，例如：

"Image Registry has been removed.**ImageStreamTags**, **BuildConfigs** and **DeploymentConfigs** which reference **ImageStreamTags** may not work as expected.Please configure storage and update the config to **Managed** state by editing configs.imageregistry.operator.openshift.io."

11.6.15.2. 镜像 registry 存储配置

对于不提供默认存储的平台，Image Registry Operator 最初将不可用。安装后，您必须配置 registry 使用的存储，这样 Registry Operator 才可用。

示配置生产集群所需的持久性卷的说明。如果适用，显示有关将空目录配置为存储位置的说明，该位置只可用于非生产集群。

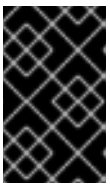
另外还提供了在升级过程中使用 **Recreate** rollout 策略来允许镜像 registry 使用块存储类型的说明。

11.6.15.2.1. 为 VMware vSphere 配置 registry 存储

作为集群管理员，在安装后需要配置 registry 来使用存储。

先决条件

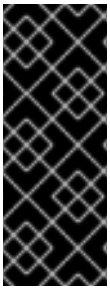
- 具有 Cluster Administrator 权限
- VMware vSphere上有一个集群。
- 为集群置备的持久性存储，如 Red Hat OpenShift Container Storage。



重要

如果您只有一个副本，OpenShift Container Platform 支持对镜像 registry 存储的 **ReadWriteOnce** 访问。要部署支持高可用性的、带有两个或多个副本的镜像 registry，需要 **ReadWriteMany** 访问设置。

- 必须有“100Gi”容量。



重要

测试显示，在 RHEL 中使用 NFS 服务器作为核心服务的存储后端可能会出现问题。这包括 OpenShift Container Registry 和 Quay，Prometheus 用于监控存储，以及 Elasticsearch 用于日志存储。因此，不推荐使用 RHEL NFS 作为 PV 后端用于核心服务。

市场上的其他 NFS 实现可能没有这些问题。如需了解更多与此问题相关的信息，请联络相关的 NFS 厂商。

流程

1. 为了配置 registry 使用存储，需要修改 **configs.imageregistry/cluster** 资源中的 **spec.storage.pvc** 。

注意

使用共享存储时，请查看您的安全设置以防止被外部访问。

2. 验证您没有 registry pod:

```
$ oc get pod -n openshift-image-registry
```

注意

如果存储类型为 **emptyDIR**，则副本数不能超过 **1**。

3. 检查 registry 配置：

```
$ oc edit configs.imageregistry.operator.openshift.io
```

输出示例

```
storage:
  pvc:
    claim: 1
```

1 将 **claim** 字段留空以允许自动创建一个 **image-registry-storage** PVC。

4. 检查 **clusteroperator** 的状态：

```
$ oc get clusteroperator image-registry
```

11.6.16. OpenShift Container Platform 的 Telemetry 访问

在 OpenShift Container Platform 4.6 中，默认运行的 Telemetry 服务提供有关集群健康状况和成功更新的指标，需要访问互联网。如果您的集群连接到互联网，Telemetry 会自动运行，而且集群会注册到 OpenShift Cluster Manager。

确认 OpenShift Cluster Manager 清单正确后，可以由 Telemetry 自动维护，也可以使用 OpenShift Cluster Manager 手动维护，使用订阅监控来跟踪帐户或多集群级别的 OpenShift Container Platform 订阅。

其他资源

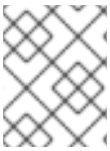
- 有关 Telemetry 服务的更多信息，请参阅关于远程健康监控。

11.6.17. 后续步骤

- 自定义集群。
- 如果需要，您可以选择不使用远程健康报告。
- 设置 registry 并配置 registry 存储。

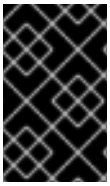
11.7. 在带有用户置备的受限网络中的 VSPHERE 上安装集群

在 OpenShift Container Platform 版本 4.6 中，您可以在受限网络中置备的 VMware vSphere 基础架构上安装集群。



注意

OpenShift Container Platform 支持将集群部署到单个 VMware vCenter 中。不支持在多个 vCenter 上使用机器/机器集部署集群。



重要

进行用户置备的基础架构安装的步骤仅作为示例。使用您提供的基础架构安装集群需要了解 vSphere 平台和 OpenShift Container Platform 的安装过程。使用用户置备的基础架构安装说明作为指南；您可以通过其他方法创建所需的资源。

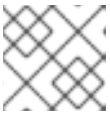
11.7.1. 先决条件

- 在镜像主机上创建镜像 registry，并获取您的 OpenShift Container Platform 版本的 **imageContentSources** 数据。

重要

由于安装介质位于堡垒主机上，因此请使用该计算机完成所有安装步骤。

- 为集群置备持久性存储。若要部署私有镜像 registry，您的存储必须提供 **ReadWriteMany** 访问模式。
- 查看有关 OpenShift Container Platform 安装和更新流程的详细信息。
- 完成安装要求您在 vSphere 主机上上传 Red Hat Enterprise Linux CoreOS(RHCOS)OVA。完成此过程的机器需要访问 vCenter 和 ESXi 主机上的端口 443。您确认可以访问端口 443。
- 如果您使用防火墙，您与管理员确认可以访问端口 443。control plane 节点必须能够通过端口 443 访问 vCenter 和 ESXi 主机，才能成功安装。
- 如果使用防火墙并计划使用遥测（telemetry），您必须将防火墙配置为允许集群需要访问的站点。



注意

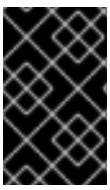
如果您要配置代理，请务必也要查看此站点列表。

11.7.2. 关于在受限网络中安装

在 OpenShift Container Platform 4.6 中，可以执行不需要有效的互联网连接来获取软件组件的安装。受限网络安装可使用安装程序置备的基础架构或用户置备的基础架构完成，具体取决于您要安装集群的云平台。

如果选择在云平台中执行受限网络安装，仍然需要访问其云 API。有些云功能，比如 Amazon Web Service 的 Route 53 DNS 和 IAM 服务，需要访问互联网。根据您的网络，在裸机硬件或 VMware vSphere 上安装时可能需要较少的互联网访问。

要完成受限网络安装，您必须创建一个 registry，镜像 OpenShift Container Platform registry 的内容并包含其安装介质。您可以在堡垒主机上创建此镜像，该主机可同时访问互联网和您的封闭网络，也可以使用满足您的限制条件的其他方法。



重要

由于用户置备安装配置的复杂性，在尝试使用用户置备的基础架构受限网络安装前，请考虑完成标准用户置备的基础架构安装。通过完成此测试安装，您可以更轻松地隔离和排查您在受限网络中安装时可能出现的问题。

11.7.2.1. 其他限制

受限网络中的集群还有以下额外限制：

- **ClusterVersion** 状态包含一个 **Unable to retrieve available updates** 错误。
- 默认情况下，您无法使用 Developer Catalog 的内容，因为您无法访问所需的镜像流标签。

11.7.3. OpenShift Container Platform 的互联网访问

在 OpenShift Container Platform 4.6 中，您需要访问互联网来获得用来安装集群的镜像。

您必须具有以下互联网访问权限：

- 访问 OpenShift Cluster Manager 以下载安装程序并执行订阅管理。如果集群可以访问互联网，并且没有禁用 Telemetry，该服务会自动授权您的集群。

- 访问 Quay.io，以获取安装集群所需的软件包。
- 获取执行集群更新所需的软件包。

重要

如果您的集群无法直接访问互联网，则可以在置备的某些类基础架构上执行受限网络安装。在此过程中，您要下载所需的内容，并使用它在镜像 registry（mirror registry） 中填充安装集群并生成安装程序所需的软件包。对于某些安装类型，集群要安装到的环境不需要访问互联网。在更新集群之前，要更新 registry 镜像系统中的内容。

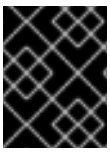
11.7.4. VMware vSphere 基础架构要求

您必须在满足您使用的组件要求的 VMware vSphere 版本 6 或 7 实例上安装 OpenShift Container Platform 集群。

表 11.62. VMware 组件支持的最低 vSphere 版本

组件	最低支持版本	描述
虚拟机监控程序	vSphere 6.5 及之后的版本 13	此版本是 Red Hat Enterprise Linux CoreOS(RHCOS)支持的最低版本。请查看 Red Hat Enterprise Linux 8 支持的管理程序列表。
使用 in-tree 驱动程序存储	vSphere 6.5 及之后的版本	此插件使用 OpenShift Container Platform 中包含的 vSphere 的树内存储驱动程序创建 vSphere 存储。
可选： Networking (NSX-T)	vSphere 6.5U3 或 vSphere 6.7U2 及之后的版本	OpenShift Container Platform 需要 vSphere 6.5U3 或 vSphere 6.7U2+。VMware 的 NSX Container Plug-in（NCP）3.0.2 使用 OpenShift Container Platform 4.6 和 NSX-T 3.x+ 认证。

如果您使用 vSphere 版本 6.5 实例，请在安装 OpenShift Container Platform 前考虑升级到 6.7U3 或 7.0。



重要

您必须确保在安装 OpenShift Container Platform 前同步 ESXi 主机上的时间。请参阅 VMware 文档中的编辑主机时间配置。

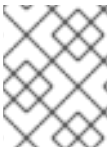
11.7.5. 具有用户置备基础架构的集群的机器要求

对于含有用户置备的基础架构的集群，您必须部署所有所需的机器。

11.7.5.1. 所需的机器

最小的 OpenShift Container Platform 集群需要下列主机：

- 一个临时 bootstrap 机器
- 三台 control plane 或 master 机器
- 至少两台计算机器，也称为 worker 机器。



注意

集群要求 bootstrap 机器在三台 control plane 机器上部署 OpenShift Container Platform 集群。您可在安装集群后删除 bootstrap 机器。

重要

要保持集群的高可用性，请将独立的物理主机用于这些集群机器。

bootstrap 和 control plane 机器必须使用 Red Hat Enterprise Linux CoreOS (RHCOS) 作为操作系统。但是，计算机器可以在 Red Hat Enterprise Linux CoreOS(RHCOS)或 Red Hat Enterprise Linux(RHEL)7.9 之间进行选择。

请注意，RHCOS 基于 Red Hat Enterprise Linux（RHEL） 8，并继承其所有硬件认证和要求。请查看Red Hat Enterprise Linux 技术功能及限制 。

重要

所有虚拟机必须位于与安装程序相同的数据存储中。

11.7.5.2. 网络连接要求

所有 Red Hat Enterprise Linux CoreOS (RHCOS) 机器在启动过程中需要 **initramfs** 中的网络从 Machine Config Server 获取 Ignition 配置文件。在初次启动过程中，需要一个 DHCP 服务器或设置了静态 IP 地址来建立网络连接，以下载它们的 Ignition 配置文件。另外，集群中的每个 OpenShift Container Platform 节点都必须有权访问网络时间协议（NTP）服务器。如果 DHCP 服务器提供 NTP 服务器信息，Red Hat Enterprise Linux CoreOS（RHCOS）机器上的 chrony 时间服务会读取信息，并可与 NTP 服务器同步时钟。

11.7.5.3. 最低资源要求

每台集群机器都必须满足以下最低要求：

表 11.63. 最低资源要求

机器	操作系统	vCPU [1]	虚拟内存	存储	IOPS [2]
bootstrap	RHCOS	4	16 GB	100 GB	300
Control plane	RHCOS	4	16 GB	100 GB	300
Compute	RHCOS 或 RHEL 7.9	2	8 GB	100 GB	300

1. 当未启用并发多线程 (SMT) 或超线程时，一个 vCPU 相当于一个物理内核。启用后，使用以下公式来计算对应的比例：（每个内核数的线程）× sockets = vCPU。

2. OpenShift Container Platform 和 Kubernetes 对磁盘性能非常敏感，建议使用更快的存储速度，特别是 control plane 节点上需要 10 ms p99 fsync 持续时间的 etcd。请注意，在许多云平台上，存储大小和 IOPS 可一起扩展，因此您可能需要过度分配存储卷来获取足够的性能。

11.7.5.4. 证书签名请求管理

在使用您置备的基础架构时，集群只能有限地访问自动机器管理，因此您必须提供一种在安装后批准集群证书签名请求 (CSR) 的机制。**kube-controller-manager** 只能批准 kubelet 客户端 CSR。 **machine-approver** 无法保证使用 kubelet 凭证请求的提供证书的有效性，因为它不能确认是正确的机器发出了该请求。您必须决定并实施一种方法，以验证 kubelet 提供证书请求的有效性并进行批准。

11.7.6. 创建用户置备的基础架构

在部署采用用户置备的基础架构的 OpenShift Container Platform 集群前，您必须创建底层基础架构。

先决条件

- 在为集群创建支持基础架构之前，请参阅OpenShift Container Platform 4.x Tested Integrations页。

流程

1. 在每个节点上配置 DHCP 或设置静态 IP 地址。
2. 提供所需的负载均衡器。
3. 配置机器的端口。
4. 配置 DNS。
5. 确保网络可以正常工作。

11.7.6.1. 用户置备的基础架构对网络的要求

所有 Red Hat Enterprise Linux CoreOS（RHCOS）机器在启动过程中需要 **initramfs** 中的网络从机器配置服务器获取 Ignition 配置。

在初次启动过程中，需要一个 DHCP 服务器或集群中的每个机器都设置了静态 IP 地址来建立网络连接，以下载它们的 Ignition 配置文件。

建议您使用 DHCP 服务器为集群进行长期机器管理。确保 DHCP 服务器已配置为向集群机器提供持久 IP 地址和主机名。

Kubernetes API 服务器必须能够解析集群机器的节点名称。如果 API 服务器和 worker 节点位于不同的区域中，您可以配置默认 DNS 搜索区域，以便 API 服务器能够解析节点名称。另一种支持的方法是始终在节点对象和所有 DNS 请求中使用完全限定域名来指代主机。

您必须配置机器间的网络连接，以便集群组件进行通信。每台机器都必须能够解析集群中所有其他机器的主机名。

表 11.64. 所有机器到所有机器

协议	端口	描述
ICMP	N/A	网络可访问性测试
TCP	**1936**	指标
	9000-9999	主机级别的服务，包括端口 **9100-9101** 上的节点导出器和端口 **9099** 上的 Cluster Version Operator。
	10250-10259	Kubernetes 保留的默认端口
	10256	openshift-sdn
UDP	**4789**	VXLAN 和 Geneve
	6081	VXLAN 和 Geneve
	9000-9999	主机级别的服务，包括端口 **9100-9101** 上的节点导出器。
TCP/UDP	**30000-32767**	Kubernetes 节点端口

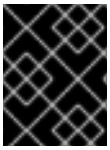
表 11.65. 要通过控制平面的所有机器

协议	端口	描述
TCP	**6443**	Kubernetes API

表 11.66. control plane 机器到 control plane 机器

协议	端口	描述
TCP	**2379-2380**	etcd 服务器和对等端口

网络拓扑要求
您为集群置备的基础架构必须满足下列网络拓扑要求。



重要

OpenShift Container Platform 要求所有节点都能访问互联网，以便为平台容器提取镜像并向红帽提供遥测数据。

负载均衡器
在安装 OpenShift Container Platform 前，您必须置备两个满足以下要求的负载均衡器：

1. **API 负载均衡器**：提供一个通用端点，供用户（包括人和机器）与平台交互和配置。配置以下条件：

- 只适用于第 4 层负载均衡。这可被称为 Raw TCP、SSL Passthrough 或者 SSL 桥接模式。如果使用 SSL Bridge 模式，必须为 API 路由启用 Server Name Indication（SNI）。
- 无状态负载平衡算法。这些选项根据负载均衡器的实现而有所不同。

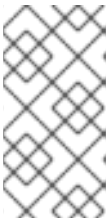
重要

不要为 API 负载均衡器配置会话持久性。

在负载均衡器的前端和后台配置以下端口：

表 11.67. API 负载均衡器

端口	后端机器（池成员）	内部	外部	描述
6443	Bootstrap 和 control plane.bootstrap 机器初始化集群 control plane 后，您要从负载均衡器中删除 bootstrap 机器。您必须为 API 服务器健康检查探测配置 **/readyz** 端点。	X	X	Kubernetes API 服务器
22623	Bootstrap 和 control plane.bootstrap 机器初始化集群 control plane 后，您要从负载均衡器中删除 bootstrap 机器。	X		机器配置服务器



注意

负载均衡器必须配置为，从 API 服务器关闭 **/readyz** 端点到从池中删除 API 服务器实例时最多需要 30 秒。在 **/readyz** 返回错误或处于健康状态后的时间范围内，端点必须被删除或添加。每 5 秒或 10 秒探测一次，有两个成功请求处于健康状态，三个成为不健康的请求经过测试。

2. **应用程序入口负载均衡器**:提供来自集群外部的应用程序流量流量的 Ingress 点。配置以下条件：
   - 只适用于第 4 层负载均衡。这可被称为 Raw TCP、SSL Passthrough 或者 SSL 桥接模式。如果使用 SSL Bridge 模式，您必须为 Ingress 路由启用Server Name Indication（SNI）。
   - 建议根据可用选项以及平台上托管的应用程序类型，使用基于连接的或者基于会话的持久性。

在负载均衡器的前端和后台配置以下端口：

表 11.68. 应用程序入口负载均衡器

端口	后端机器（池成员）	内部	外部	描述
443	默认运行入口路由器 Pod、计算或 worker 的机器。	X	X	HTTPS 流量
80	默认运行入口路由器 Pod、计算或 worker 的机器。	X	X	HTTP 流量

提示

如果负载均衡器可以看到客户端的真实 IP 地址，启用基于 IP 的会话持久性可提高使用端到端 TLS 加密的应用程序的性能。

注意

OpenShift Container Platform 集群需要正确配置入口路由器。control plane 初始化后，您必须配置入口路由器。

以太网适配器硬件地址要求
当为集群置备虚拟机时，为每个虚拟机配置的以太网接口必须使用 VMware 机构唯一识别符 (OUI) 分配范围中的 MAC 地址：

- **00:05:69:00:00:00** 到 **00:05:69:FF:FF:FF**
- **00:0c:29:00:00:00** 到 **00:0c:29:FF:FF:FF**
- **00:1c:14:00:00:00** 到 **00:1c:14:FF:FF:FF**
- **00:50:56:00:00:00** 到 **00:50:56:FF:FF:FF**

如果使用 VMware OUI 以外的 MAC 地址，集群安装将无法成功。

NTP 配置
OpenShift Container Platform 集群默认配置为使用公共网络时间协议（NTP）服务器。如果要使用本地企业 NTP 服务器，或者集群部署在断开连接的网络中，您可以将集群配置为使用特定的时间服务器。如需更多信息，请参阅*配置 chrony 时间服务*的文档。

如果 DHCP 服务器提供 NTP 服务器信息，Red Hat Enterprise Linux CoreOS（RHCOS）机器上的 chrony 时间服务会读取信息，并可与 NTP 服务器同步时钟。

其他资源

- 配置 chrony 时间服务

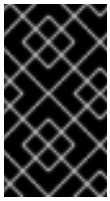
11.7.6.2. 用户置备 DNS 要求

DNS 用于名称解析和反向名称解析。DNS A/AAAA 或 CNAME 记录用于名称解析，PTR 记录用于反向解析名称。反向记录很重要，因为 Red Hat Enterprise Linux CoreOS（RHCOS）使用反向记录为所有节点设置主机名。另外，反向记录用于生成 OpenShift Container Platform 需要操作的证书签名请求（CSR）。

采用用户置备的基础架构的 OpenShift Container Platform 集群需要以下 DNS 记录。在每一记录中，**<cluster_name>** 是集群名称，**<base_domain>** 则是您在 **install-config.yaml** 文件中指定的集群基域。完整的 DNS 记录采用如下格式: **<component>.<cluster_name>.<base_domain>.**。

表 11.69. 所需的 DNS 记录

组件	记录	描述
Kubernetes API	**api.<cluster_name>.<base_domain>.**	添加 DNS A/AAAA 或 CNAME 记录，以及 DNS PTR 记录，以识别 control plane 机器的负载均衡器。这些记录必须由集群外的客户端以及集群中的所有节点解析。

组件	记录	描述
	api-int.<cluster_name>. <base_domain>.	添加 DNS A/AAAA 或 CNAME 记录，以及 DNS PTR 记录，以识别 control plane 机器的负载均衡器。这些记录必须可以从集群中的所有节点解析。  **重要** API 服务器必须能够根据在 Kubernetes 中记录的主机名解析 worker 节点。如果 API 服务器无法解析节点名称，则代理的 API 调用会失败，且您无法从 pod 检索日志。
Routes	***.apps.<cluster_name>. <base_domain>.**	添加通配符 DNS A/AAAA 或 CNAME 记录，指向以运行入口路由器 Pod 的机器（默认为 worker 节点）为目标的负载均衡器。这些记录必须由集群外的客户端以及集群中的所有节点解析。
bootstrap	**bootstrap.<cluster_name>. <base_domain>.**	添加 DNS A/AAAA 或 CNAME 记录，以及 DNS PTR 记录来识别 bootstrap 机器。这些记录必须由集群中的节点解析。
Master 主机	**<master><n>. <cluster_name>. <base_domain>.**	DNS A/AAAA 或 CNAME 记录，以识别 control plane 节点（也称为 master 节点）的每台机器。这些记录必须由集群中的节点解析。
Worker 主机	**<worker><n>. <cluster_name>. <base_domain>.**	添加 DNS A/AAAA 或 CNAME 记录，以识别 worker 节点的每台机器。这些记录必须由集群中的节点解析。

提示

您可以使用 **nslookup <hostname>** 命令来验证名称解析。您可以使用 **dig -x <ip_address>** 命令来验证 PTR 记录的反向名称解析。

下面的 BIND 区文件的例子展示了关于名字解析的 A 记录的例子。这个示例的目的是显示所需的记录。这个示例不是为选择一个名称解析服务提供建议。

例 11.13. DNS 区数据库示例

```
$TTL 1W
@ IN SOA ns1.example.com. root (
   2019070700 ; serial
   3H  ; refresh (3 hours)
   30M  ; retry (30 minutes)
   2W  ; expiry (2 weeks)
   1W )  ; minimum (1 week)
 IN NS ns1.example.com.
 IN MX 10 smtp.example.com.
;
;
```

```
ns1 IN A 192.168.1.5
smtp IN A 192.168.1.5
;
helper IN A 192.168.1.5
helper.ocp4 IN A 192.168.1.5
;
; The api identifies the IP of your load balancer.
api.ocp4  IN A 192.168.1.5
api-int.ocp4  IN A 192.168.1.5
;
; The wildcard also identifies the load balancer.
*.apps.ocp4  IN A 192.168.1.5
;
; Create an entry for the bootstrap host.
bootstrap.ocp4 IN A 192.168.1.96
;
; Create entries for the master hosts.
master0.ocp4  IN A 192.168.1.97
master1.ocp4  IN A 192.168.1.98
master2.ocp4  IN A 192.168.1.99
;
; Create entries for the worker hosts.
worker0.ocp4  IN A 192.168.1.11
worker1.ocp4  IN A 192.168.1.7
;
;EOF
```

下面的 BIND 区文件示例显示了反向名字解析的 PTR 记录示例。

例 11.14. 反向记录的 DNS 区数据库示例

```
$TTL 1W
@ IN SOA ns1.example.com. root (
   2019070700 ; serial
   3H  ; refresh (3 hours)
   30M  ; retry (30 minutes)
   2W  ; expiry (2 weeks)
   1W )  ; minimum (1 week)
 IN NS ns1.example.com.
;
; The syntax is "last octet" and the host must have an FQDN
; with a trailing dot.
97 IN PTR master0.ocp4.example.com.
98 IN PTR master1.ocp4.example.com.
99 IN PTR master2.ocp4.example.com.
;
96 IN PTR bootstrap.ocp4.example.com.
;
5 IN PTR api.ocp4.example.com.
5 IN PTR api-int.ocp4.example.com.
;
11 IN PTR worker0.ocp4.example.com.
```

```
7 IN PTR worker1.ocp4.example.com.
;
;EOF
```

11.7.7. 生成 SSH 私钥并将其添加到代理中

如果要在集群上执行安装调试或灾难恢复，则必须为 **ssh-agent** 和安装程序提供 SSH 密钥。您可以使用此密钥访问公共集群中的 bootstrap 机器来排除安装问题。

注意

在生产环境中，您需要进行灾难恢复和调试。

您可以使用此密钥以 **core** 用户身份通过 SSH 连接到 master 节点。在部署集群时，此密钥会添加到 **core** 用户的 **~/.ssh/authorized_keys** 列表中。

注意

您必须使用一个本地密钥，而不要使用在特定平台上配置的密钥，如 AWS 密钥对。

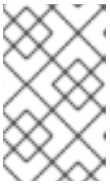
流程

1. 如果还没有为计算机上免密码身份验证而配置的 SSH 密钥，请创建一个。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ ssh-keygen -t ed25519 -N '' \
  -f <path>/<file_name> 1
```

1 指定新 SSH 密钥的路径和文件名，如 **~/.ssh/id_rsa**。如果您已有密钥对，请确保您的公钥位于 **~/.ssh** 目录中。

运行此命令会在指定的位置生成不需要密码的 SSH 密钥。



注意

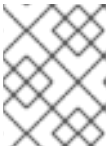
如果您计划在 **x86_64** 架构中安装使用 FIPS 验证的/Modules in Process 加密库的 OpenShift Container Platform 集群，不要创建使用 **ed25519** 算法的密钥。反之，创建一个使用 **rsa** 或 **ecdsa** 算法的密钥。

2. 作为后台任务启动 **ssh-agent** 进程：

```
$ eval "$(ssh-agent -s)"
```

输出示例

```
Agent pid 31874
```



注意

如果您的集群采用 FIPS 模式，则只使用 FIPS 兼容算法来生成 SSH 密钥。密钥必须是 RSA 或 ECDSA。

3. 将 SSH 私钥添加到 **ssh-agent**：

```
$ ssh-add <path>/<file_name> 1
```

输出示例

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

1 指定 SSH 私钥的路径和文件名，如 **~/.ssh/id_rsa**

后续步骤

- 在安装 OpenShift Container Platform 时，为安装程序提供 SSH 公钥。如果在您置备的基础架构上安装集群，您必须将此密钥提供给集群的机器。

11.7.8. 手动创建安装配置文件

对于使用用户置备的基础架构的 OpenShift Container Platform 安装，您必须手动生成安装配置文件。

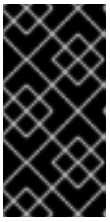
先决条件

- 获取 OpenShift Container Platform 安装程序和集群的访问令牌。
- 获取命令输出中的 **imageContentSources** 部分来镜像存储库。
- 获取您的镜像 registry 的证书内容。

流程

1. 创建用来存储您所需的安装资产的安装目录：

```
$ mkdir <installation_directory>
```



重要

您必须创建目录。一些安装信息，如 bootstrap X.509 证书，有较短的过期间隔，因此不要重复使用安装目录。如果要重复使用另一个集群安装中的个别文件，可以将其复制到您的目录中。但是，一些安装数据的文件名可能会在发行版本之间有所改变。从 OpenShift Container Platform 老版本中复制安装文件时要格外小心。

2. 自定义以下 **install-config.yaml** 文件模板，并将它保存到 **<installation_directory>** 中。

注意

此配置文件必须命名为 **install-config.yaml**。

- 除非使用 RHCOS 默认信任的 registry，如 **docker.io**，否则必须在 **additionalTrustBundle** 部分中提供镜像存储库的证书内容。在大多数情况下，必须为您的镜像提供证书。
- 您必须包含命令输出中的 **imageContentSources** 部分，才能镜像存储库。

3. 备份 **install-config.yaml** 文件，以便用于安装多个集群。

重要

install-config.yaml 文件会在安装过程的下一步骤中消耗掉。现在必须备份它。

11.7.8.1. VMware vSphere install-config.yaml 文件示例

您可以自定义 **install-config.yaml** 文件，以指定有关 OpenShift Container Platform 集群平台的更多信息，或修改所需参数的值。

```
apiVersion: v1
baseDomain: example.com (1)
compute:
- hyperthreading: Enabled (2) (3)
  name: worker
  replicas: 0 (4)
controlPlane:
  hyperthreading: Enabled (5) (6)
  name: master
  replicas: 3 (7)
metadata:
  name: test (8)
platform:
  vsphere:
    vcenter: your.vcenter.server (9)
    username: username (10)
    password: password (11)
    datacenter: datacenter (12)
    defaultDatastore: datastore (13)
    folder: "/<datacenter_name>/vm/<folder_name>/<subfolder_name>" (14)
fips: false (15)
pullSecret: '{"auths":{"<local_registry>": {"auth": "<credentials>","email": "you@example.com"}}}' (16)
sshKey: 'ssh-ed25519 AAAA...' (17)
additionalTrustBundle: | (18)
  -----BEGIN CERTIFICATE-----
  ZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZ
  -----END CERTIFICATE-----
imageContentSources: (19)
- mirrors:
  - <local_registry>/<local_repository_name>/release
  source: quay.io/openshift-release-dev/ocp-release
- mirrors:
  - <local_registry>/<local_repository_name>/release
  source: quay.io/openshift-release-dev/ocp-v4.0-art-dev
```

(1) 集群的基域。所有 DNS 记录都必须是这个基域的子域，并包含集群名称。

2 5 **controlPlane** 部分是一个单映射，但 compute 部分是一系列映射。为满足不同数据结构的要求，**compute** 部分的第一行必须以连字符 **-** 开头，**controlPlane** 部分的第一行则不可以连字符开

3 6 是否要启用或禁用并发多线程或**超线程**。默认情况下，启用并发多线程以提高机器内核的性能。您可以通过将参数值设为 **Disabled** 来禁用。如果您在某些集群机器上禁用并发多线程，则必须在所有集群机器上禁用。

重要

如果禁用并发多线程，请确保在容量规划时考虑到机器性能可能会显著降低的问题。如果您禁用并发多线程，则计算机必须至少使用 8 个 CPU 和 32GB RAM。

4 **replicas** 参数的值必须设置为 **0**。此参数控制集群为您创建和管理的 worker 数量，使用用户置备的基础架构时集群不会执行这些功能。在完成 OpenShift Container Platform 安装前，您必须手动为集群部署 worker 机器。

7 您添加到集群的 control plane 机器数量。由于集群将这个值用作集群中 etcd 端点的数量，因此该值必须与您部署的 control plane 机器数量匹配。

8 您在 DNS 记录中指定的集群名称。

9 vCenter 服务器的完全限定主机名或 IP 地址。

10 用于访问服务器的用户名。此用户必须至少具有 vSphere 中 静态或动态持久性卷置备 所需的角色和权限。

11 与 vSphere 用户关联的密码。

12 vSphere 数据中心。

13 要使用的默认 vSphere 数据存储。

14 可选：对于安装程序置备的基础架构，安装程序创建虚拟机的现有文件夹的绝对路径，如 **/<datacenter_name>/vm/<folder_name>/<subfolder_name>**。如果没有提供这个值，安装程序会在数据中心虚拟机文件夹中创建一个顶层文件夹，其名称为基础架构 ID。如果您为集群提供基础架构,请省略此参数。

15 是否启用或禁用 FIPS 模式。默认情况下不启用 FIPS 模式。如果启用了 FIPS 模式，运行 OpenShift Container Platform 的 Red Hat Enterprise Linux CoreOS (RHCOS) 机器会绕过默认的 Kubernetes 加密套件，并使用由 RHCOS 提供的加密模块。

重要

只有在 **x86_64** 架构中的 OpenShift Container Platform 部署支持 FIPS 验证的/Modules in Process 加密库。

16 对于 **<local_registry>**，请指定 registry 域名，以及您的镜像 registry 用来提供内容的可选端口。例如： **registry.example.com** 或者 **registry.example.com:5000**。使用 **<credentials>** 为您生成的镜像 registry 指定 base64 编码的用户名和密码。

17 Red Hat Enterprise Linux CoreOS (RHCOS) 中 **core** 用户的默认 SSH 密钥的公钥部分。

注意

对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 **ssh-agent** 进程使用的 SSH 密钥。

18 提供用于镜像 registry 的证书文件内容。

19 提供命令输出中的 **imageContentSources** 部分来镜像存储库。

11.7.8.2. 在安装过程中配置集群范围代理

生产环境可能会拒绝直接访问互联网，而是提供 HTTP 或 HTTPS 代理。您可以通过在 **install-config.yaml** 文件中配置代理设置，将新的 OpenShift Container Platform 集群配置为使用代理。

先决条件

- 您有一个现有的 **install-config.yaml** 文件。
- 您检查了集群需要访问的站点，并决定是否需要绕过代理。默认情况下代理所有集群出口流量，包括对托管云供应商 API 的调用。您需要将站点添加到 **Proxy** 对象的 **spec.noProxy** 字段来绕过代理。

注意

Proxy 对象 **status.noProxy** 字段使用安装配置中的 **networking.machineNetwork[].cidr**、**networking.clusterNetwork[].cidr** 和 **networking.serviceNetwork[]** 字段的值填充。

对于在 Amazon Web Services(AWS)、Google Cloud Platform(GCP)、Microsoft Azure 和 Red Hat OpenStack Platform(RHOSP)上安装, **Proxy** 对象 **status.noProxy** 字段也会使用实例元数据端点填充(**169.254.169.254**)。

流程

1. 编辑 **install-config.yaml** 文件并添加代理设置。例如：

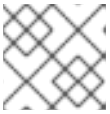
```
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: example.com 3
additionalTrustBundle: | 4
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
...
```

1 用于创建集群外 HTTP 连接的代理 URL。URL 必须是 **http**。

2 用于创建集群外 HTTPS 连接的代理 URL。

3 要排除在代理中的目标域名、IP 地址或其他网络 CIDR 的逗号分隔列表。在域前面加 **.** 来仅

4 如果提供，安装程序会在 **openshift-config 命名空间中生成名为 user-ca- bundle** 的配置映射来保存额外的 CA 证书。如果您提供 **additionalTrustBundle** 和至少一个代理设置，则



注意

安装程序不支持代理的 **readinessEndpoints** 字段。

2. 保存该文件，并在安装 OpenShift Container Platform 时引用。

安装程序会创建一个名为 **cluster** 的集群范围代理，该代理使用提供的 **install-config.yaml** 文件中的代理设置。如果没有提供代理设置，仍然会创建一个 **cluster Proxy** 对象，但它会有一个空 **spec**。

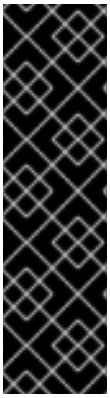
注意

只支持名为 **cluster** 的 **Proxy** 对象，且无法创建额外的代理。

11.7.9. 创建 Kubernetes 清单和 Ignition 配置文件

由于您必须修改一些集群定义文件并要手动启动集群机器，因此您必须生成 Kubernetes 清单和 Ignition 配置文件，集群需要这两项来创建其机器。

安装配置文件转换为 Kubernetes 清单。清单嵌套到 Ignition 配置文件中，稍后用于创建集群。



重要

- 安装程序生成的 Ignition 配置文件包含在 24 小时后过期的证书，然后在过期时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外情况是，您需要手动批准待处理的 **node-bootstrapper** 证书签名请求（CSR）来恢复 kubelet 证书。如需更多信息，请参阅 *从过期的 control plane 证书中恢复*的文档。
- 建议您在生成 12 小时后使用 Ignition 配置文件，因为集群安装后 24 小时证书从 16 小时轮转至 22 小时。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中运行证书更新时避免安装失败。

先决条件

- 已获得 OpenShift Container Platform 安装程序。对于受限网络安装，这些文件位于您的堡垒主机上。
- 已创建 **install-config.yaml** 安装配置文件。

流程

1. 切换到包含安装程序的目录，并为集群生成 Kubernetes 清单：

```
$ ./openshift-install create manifests --dir <installation_directory> 1
```

1 对于 **<installation_directory>**，请指定含有您创建的 **install-config.yaml** 文件的安装目录。

2. 删除定义 control plane 机器的 Kubernetes 清单文件以及计算机器集：

```
$ rm -f openshift/99_openshift-cluster-api_master-machines-*.yaml openshift/99_openshift-cluster-api_worker-machineset-*.yaml
```

由于您要自行创建和管理这些资源，因此不必初始化这些资源。

- 您可以使用机器 API 来保留机器集文件来创建计算机器，但您必须更新对其的引用，以匹配您的环境。

3. 检查 **<installation_directory>/manifests/cluster-scheduler-02-config.yml** Kubernetes 清单文件中的 **mastersSchedulable** 参数是否已设置为 **false**。此设置可防止在 control plane 机器上调度 pod:
   a. 打开 **<installation_directory>/manifests/cluster-scheduler-02-config.yml** 文件。
   b. 找到 **mastersSchedulable** 参数并确保它被设置为 **false**。
   c. 保存并退出文件。
4. 要创建 Ignition 配置文件，从包含安装程序的目录运行以下命令：

```
$ ./openshift-install create ignition-configs --dir <installation_directory> 1
```

1 对于 **<installation_directory>**，请指定相同的安装目录。

该目录中将生成以下文件：

```
.
├── auth
│   ├── kubeadmin-password
│   └── kubeconfig
├── bootstrap.ign
├── master.ign
├── metadata.json
└── worker.ign
```

11.7.10. 配置 chrony 时间服务

您需要修改 **chrony.conf** 文件的内容来设置 chrony 时间服务（**chronyd**）使用的时间服务器和相关设置，并通过一个机器配置将这些内容传递给节点。

流程

1. 创建 **chrony.conf** 文件的内容并对其进行 base64 编码。例如：

```
$ cat << EOF | base64
   pool 0.rhel.pool.ntp.org iburst 1
   driftfile /var/lib/chrony/drift
   makestep 1.0 3
   rtcsync
   logdir /var/log/chrony
EOF
```

1 指定任何有效的、可访问的时间源，如 DHCP 服务器提供的时间源。

输出示例

```
ICAgIHNlcnZlciBjbG9jay5yZWRoYXQuY29tIGlidXJzdAogICAgZHJpZnRmaWxlIC92YXIvbGli
L2Nocm9ueS9kcmlmdAogICAgbWFrZXN0ZXAgMS4wIDMKICAgIHJ0Y3N5bmMKICAgIGxvZ2
RpciAv
dmFyL2xvZy9jaHJvbnkK
```

2. 创建 **MachineConfig** 对象文件，将 base64 字符串替换为您刚刚创建的字符串。本例将文件添加到 **master** 节点。您可以将其更改为 **worker**，或为 **worker** 角色创建额外的 MachineConfig。为集群使用的每种机器创建 MachineConfig 文件：

```
$ cat << EOF > ./99-masters-chrony-configuration.yaml
apiVersion: machineconfiguration.openshift.io/v1
kind: MachineConfig
metadata:
  labels:
    machineconfiguration.openshift.io/role: master
  name: 99-masters-chrony-configuration
spec:
  config:
    ignition:
      config: {}
      security:
        tls: {}
      timeouts: {}
      version: 3.1.0
    networkd: {}
    passwd: {}
    storage:
      files:
      - contents:
          source: data:text/plain;charset=utf-
8;base64,ICAgIHNlcnZlciBjbG9jay5yZWRoYXQuY29tIGlidXJzdAogICAgZHJpZnRmaWxlIC92Y
XIvbGliL2Nocm9ueS9kcmlmdAogICAgbWFrZXN0ZXAgMS4wIDMKICAgIHJ0Y3N5bmMKICAg
IGxvZ2RpciAvdmFyL2xvZy9jaHJvbnkK
        mode: 420 (1)
        overwrite: true
        path: /etc/chrony.conf
  osImageURL: ""
EOF
```

(1) 为机器配置文件的 **mode** 字段指定数值模式。在创建文件并应用更改后，**模式** 将转换为十进制值。您可以使用 **oc get mc <mc-name> -o yaml** 命令来检查 YAML 文件。

3. 对配置文件做一个备份副本。

4. 使用两种方式之一应用配置：

   - 如果集群还没有启动，在生成清单文件后，将此文件添加到 **<installation_directory>/openshift** 目录中，然后继续创建集群。
   - 如果集群已在运行，请应用该文件：

     ```
     $ oc apply -f ./99-masters-chrony-configuration.yaml
     ```

11.7.11. 提取基础架构名称

Ignition 配置文件包含一个唯一的集群标识符,您可以使用它在 VMware vSphere 中唯一地标识您的集群。如果计划使用集群标识符作为虚拟机文件夹的名称,您必须提取它。

先决条件

- 获取 OpenShift Container Platform 安装程序以及集群的 pull secret。
- 已为集群生成 Ignition 配置文件。
- 安装了 **jq** 软件包。

流程

- 要从 Ignition 配置文件元数据中提取和查看基础架构名称，请运行以下命令：

```
$ jq -r .infraID <installation_directory>/metadata.json 1
```

1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

输出示例

```
openshift-vw9j6 1
```

1 此命令的输出是您的集群名称和随机字符串。

11.7.12. 在 vSphere 中创建 Red Hat Enterprise Linux CoreOS (RHCOS) 机器

在 VMware vSphere 上安装包含用户置备基础架构的集群前，您必须在 vSphere 主机上创建 RHCOS 机器供其使用。

先决条件

- 已获取集群的 Ignition 配置文件。
- 您可以从计算机访问 HTTP 服务器，并且您创建的机器也可以访问该服务器。
- 您已创建了 vSphere 集群。

流程

1. 将名为 **<installation_directory>/bootstrap.ign** 的 bootstrap Ignition 配置文件上传到 HTTP 服务器，该配置文件是由安装程序创建的。记下此文件的 URL。
2. 将 bootstrap 节点的以下辅助 Ignition 配置文件保存到计算机中，存为 **<installation_directory>/merge-bootstrap.ign** ：

```
{
  "ignition": {
    "config": {
      "merge": [
```

```
        {
          "source": "<bootstrap_ignition_config_url>", 1
          "verification": {}
        }
      ]
    },
    "timeouts": {},
    "version": "3.1.0"
  },
  "networkd": {},
  "passwd": {},
  "storage": {},
  "systemd": {}
}
```

1 指定您托管的 bootstrap Ignition 配置文件的 URL。

为 bootstrap 机器创建虚拟机 (VM) 时，您要使用此 Ignition 配置文件。

3. 找到安装程序创建的以下 Ignition 配置文件：

   - **<installation_directory>/master.ign**
   - **<installation_directory>/worker.ign**
   - **<installation_directory>/merge-bootstrap.ign**

4. 将 Ignition 配置文件转换为 Base64 编码。在此流程中，您必须将这些文件添加到虚拟机中的额外配置参数 **guestinfo.ignition.config.data** 中。
   例如，如果您使用 Linux 操作系统，可以使用 **base64** 命令来编码这些文件。

```
$ base64 -w0 <installation_directory>/master.ign > <installation_directory>/master.64
```

```
$ base64 -w0 <installation_directory>/worker.ign > <installation_directory>/worker.64
```

```
$ base64 -w0 <installation_directory>/merge-bootstrap.ign > <installation_directory>/merge-bootstrap.64
```

重要

如果您计划在安装完成后在集群中添加更多计算机器，请不要删除这些文件。

5. 获取 RHCOS OVA 镜像。镜像位于 RHCOS 镜像镜像页面。

重要

RHCOS 镜像可能不会随着 OpenShift Container Platform 的每一发行版本都有改变。您必须下载一个最高版本的镜像，其版本号应小于或等于您安装的 OpenShift Container Platform 版本。如果可用，请使用与 OpenShift Container Platform 版本匹配的镜像版本。

文件名包含 OpenShift Container Platform 版本号，格式为 **rhcos-vmware.<architecture>.ova**。

6. 在 vSphere 客户端中，在数据中心中创建一个文件夹来存储您的虚拟机。

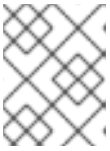
   a. 点击 **VMs and Templates** 视图。

   b. 右键点击您的数据中心名称。

   c. 点击 **New Folder → New VM and Template Folder**。

   d. 在显示的窗口中输入文件夹名称。如果您没有在 **install-config.yaml** 文件中指定现有文件夹，请创建一个文件夹，其名称与基础架构 ID 相同。您可以使用这个文件夹名称，因此 vCenter 会在适当的位置为 Workspace 配置动态置备存储。

7. 在 vSphere 客户端中，为 OVA 镜像创建一个模板，然后根据需要克隆模板。

   注意

   在以下步骤中，您将创建一个模板，然后克隆所有集群机器的模板。然后，在置备虚拟机时，为该克隆的机器类型提供 Ignition 配置文件的位置。

   a. 在 **Hosts and Clusters** 选项卡中，右键点击您的集群名称并选择 **Deploy OVF Template**。

   b. 在 **Select an OVF** 选项卡中，指定您下载的 RHCOS OVA 文件的名称。

   c. 在 **Select a name and folder** 选项卡中，为您的模板设置 **虚拟机名称**，如 **Template-RHCOS**。点击 vSphere 集群的名称并选择您在上一步中创建的文件夹。

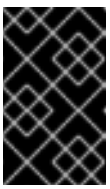
   d. 在 **Select a compute resource** 选项卡中，点击您的 vSphere 集群名称。

   e. 在 **Select storage** 选项卡中，配置虚拟机的存储选项。

      - 根据您的存储要求，选择**Thin Provision** 或 **Thick Provision**。
      - 选择您在 **install-config.yaml** 文件中指定的数据存储。

   f. 在 **Select network** 选项卡中，指定您为集群配置的网络（如果可用）。

   g. 在创建 OVF 模板时，请不要在 **Customize template** 选项卡上指定值，或者不要再配置模板。

   重要

   不要启动原始虚拟机模板。VM 模板必须保持关闭状态，必须为新的 RHCOS 机器克隆。启动虚拟机模板会将虚拟机模板配置为平台上的虚拟机，这样可防止它被用作计算机集可以应用配置的模板。

8. 部署模板后，为集群中的机器部署虚拟机。

   a. 右键点击模板的名称，再点击 **Clone → Clone to Virtual Machine**。

   b. 在 **Select a name and folder** 选项卡中，指定虚拟机的名称。名称中可以包括机器类型，如 **control-plane-0** 或 **compute-1**。

   c. 在 **Select a name and folder** 选项卡中，选择您为集群创建的文件夹名称。

d. 在 **Select a compute resource** 选项卡中，选择数据中心中的主机名称。
对于 bootstrap 机器，指定您托管的 bootstrap Ignition 配置文件的 URL。

e. 可选：在 **Select storage** 选项卡中，自定义存储选项。

f. 在 **Select clone options** 中，选择 **Customize this virtual machine's hardware**。

g. 在 **Customize hardware** 选项卡中，点击 **VM Options → Advanced**。

- 可选：覆盖 vSphere 中的默认 DHCP 网络。启用静态 IP 网络：

  i. 设置静态 IP 配置：

  ```
  $ export IPCFG="ip=<ip>::<gateway>:<netmask>:<hostname>:<iface>:none
  nameserver=srv1 [nameserver=srv2 [nameserver=srv3 [...]]]"
  ```

  示例命令

  ```
  $ export IPCFG="ip=192.168.100.101::192.168.100.254:255.255.255.0:::none
  nameserver=8.8.8.8"
  ```

  ii. 在从 vSphere 中的 OVA 引导虚拟机前，设置 **guestinfo.afterburn.initrd.network-kargs** 属性：

  ```
  $ govc vm.change -vm "<vm_name>" -e "guestinfo.afterburn.initrd.network-
  kargs=${IPCFG}"
  ```

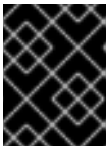
- 可选：在出现集群性能问题时，从 **Latency Sensitivity** 列表中选择 **High**。确定虚拟机的 CPU 和内存保留有以下值：
  - 内存保留值必须等于其配置的内存大小。
  - CPU 保留值必须至少是低延迟虚拟 CPU 的数量，乘以测量的物理 CPU 速度。

- 点击 **Edit Configuration**，然后在 **Configuration Parameters** 窗口中点击 **Add Configuration Params**。定义以下参数名称和值：
  - **guestinfo.ignition.config.data** ：找到您在此流程中创建的 base-64 编码文件，并粘贴此机器类型的 base64 编码 Ignition 配置文件的内容。
  - **guestinfo.ignition.config.data.encoding**：指定 **base64**。
  - **disk.EnableUUID**：指定 **TRUE**。

h. 在 **Customize hardware** 选项卡的 **Virtual Hardware** 面板中，根据需要修改指定的值。确保 RAM、CPU 和磁盘存储的数量满足机器类型的最低要求。

i. 完成配置并打开虚拟机电源。

9. 对于每台机器，按照前面的步骤为集群创建其余的机器。



重要

此刻您必须创建 bootstrap 和 control plane 机器。由于计算机器中已默认部署了一些 Pod，因此在安装集群前，还要创建至少两台计算机器。

11.7.13. 在 vSphere 中创建更多 Red Hat Enterprise Linux CoreOS（RHCOS）机器

您可以为集群创建更多计算机器，在 VMware vSphere 上使用用户置备的基础架构。

先决条件

- 获取计算机器的 Base64 编码 Ignition 文件。
- 您可以访问您为集群创建的 vSphere 模板。

流程

1. 部署模板后，为集群中的机器部署虚拟机。
   a. 右键点击模板的名称，再点击 **Clone → Clone to Virtual Machine**。
   b. 在 **Select a name and folder**选项卡中，指定虚拟机的名称。您可以在名称中包含机器类型，如 **compute-1**。
   c. 在 **Select a name and folder**选项卡中，选择您为集群创建的文件夹名称。
   d. 在 **Select a compute resource** 选项卡中，选择数据中心中的主机名称。
   e. 可选：在 **Select storage** 选项卡中，自定义存储选项。
   f. 在 **Select clone options** 中，选择 **Customize this virtual machine's hardware**。
   g. 在 **Customize hardware** 选项卡中，点击 **VM Options → Advanced**。
      - 从 **Latency Sensitivity** 列表中选择 **High**。
      - 点击 **Edit Configuration**，然后在 **Configuration Parameters** 窗口中点击 **Add Configuration Params**。定义以下参数名称和值：
        - **guestinfo.ignition.config.data**：粘贴此机器类型的 Base64 编码计算 Ignition 配置文件的内容。
        - **guestinfo.ignition.config.data.encoding**：指定 **base64**。
        - **disk.EnableUUID**：指定 **TRUE**。
   h. 在 **Customize hardware** 选项卡的 **Virtual Hardware** 面板中，根据需要修改指定的值。确保 RAM、CPU 和磁盘存储的数量满足机器类型的最低要求。另外，如果有多个可用的网络，请确定在 **Add network adapter 中** 选择正确的网络。
   i. 完成配置并打开虚拟机电源。
2. 继续为集群创建更多计算机器。

11.7.14. 磁盘分区

在大多数情况下，数据分区最初是由安装 RHCOS 而不是安装另一个操作系统来创建的。在这种情况下，OpenShift Container Platform 安装程序应该被允许配置磁盘分区。

但是，在安装 OpenShift Container Platform 节点时，在两种情况下您可能需要覆盖默认分区：

- 创建单独的分区：对于在空磁盘中的 greenfield 安装，您可能想要在分区中添加单独的存储。这只在生成 **/var** 或者一个 **/var** 独立分区的子目录（如 **/var/lib/etcd** ）时被正式支持，但不支持两者。

重要

Kubernetes 只支持两个文件系统分区。如果您在原始配置中添加多个分区，Kubernetes 无法监控所有这些分区。

- 保留现有分区：对于 brownfield 安装，您要在现有节点上重新安装 OpenShift Container Platform，并希望保留从之前的操作系统中安装的数据分区，对于 **coreos-installer** 来说，引导选项和选项都允许您保留现有数据分区。

创建一个独立的 /var 分区

通常情况下，OpenShift Container Platform 的磁盘分区应该留给安装程序。然而，在有些情况下您可能需要在文件系统的一部分中创建独立分区。

OpenShift Container Platform 支持添加单个分区来将存储附加到 **/var** 分区或 **/var** 的子目录。例如：

- **/var/lib/containers**：保存镜像相关的内容，随着更多镜像和容器添加到系统中，它所占用的存储会增加。
- **/var/lib/etcd**：保存您可能希望保持独立的数据，比如 etcd 存储的性能优化。
- **/var**：保存您希望独立保留的数据，用于特定目的（如审计）。

单独存储 **/var** 目录的内容可方便地根据需要对区域扩展存储，并可以在以后重新安装 OpenShift Container Platform 时保持该数据地完整。使用这个方法，您不必再次拉取所有容器，在更新系统时也无法复制大量日志文件。

因为 **/var** 在进行一个全新的 Red Hat Enterprise Linux CoreOS（RHCOS）安装前必需存在，所以这个流程会在 OpenShift Container Platform 安装过程的 **openshift-install** 准备阶段插入的机器配置来设置独立的 **/var** 分区。

流程

1. 创建存放 OpenShift Container Platform 安装文件的目录：

```
$ mkdir $HOME/clusterconfig
```

2. 运行 **openshift-install** 在 **manifest** 和 **openshift** 子目录中创建一组文件。在出现提示时回答系统问题：

```
$ openshift-install create manifests --dir $HOME/clusterconfig
? SSH Public Key ...
$ ls $HOME/clusterconfig/openshift/
99_kubeadmin-password-secret.yaml
99_openshift-cluster-api_master-machines-0.yaml
99_openshift-cluster-api_master-machines-1.yaml
99_openshift-cluster-api_master-machines-2.yaml
...
```

3. 创建 **MachineConfig** 对象并将其添加到 **openshift** 目录中的一个文件中。例如，把文件命名为 **98-var-partition.yaml**，将磁盘设备名称改为 **worker** 系统中存储设备的名称，并根据情况设置存储大小。这个示例将 **/var** 目录放在一个单独的分区中：

```
apiVersion: machineconfiguration.openshift.io/v1
kind: MachineConfig
metadata:
  labels:
    machineconfiguration.openshift.io/role: worker
  name: 98-var-partition
spec:
  config:
    ignition:
      version: 3.1.0
    storage:
      disks:
      - device: /dev/<device_name> 1
        partitions:
        - label: var
          startMiB: <partition_start_offset> 2
          sizeMiB: <partition_size> 3
      filesystems:
        - device: /dev/disk/by-partlabel/var
          path: /var
          format: xfs
    systemd:
      units:
        - name: var.mount 4
          enabled: true
          contents: |
            [Unit]
            Before=local-fs.target
            [Mount]
            What=/dev/disk/by-partlabel/var
            Where=/var
            Options=defaults,prjquota 5
            [Install]
            WantedBy=local-fs.target
```

1. 要分区的磁盘的存储设备名称。
2. 当在引导磁盘中添加数据分区时，推荐最少使用 25000MB。root 文件系统会自动重新定义大小使其占据所有可用空间（最多到指定的偏移值）。如果没有指定值，或者指定的值小于推荐的最小值，则生成的 root 文件系统会太小，而在以后进行的 RHCOS 重新安装可能会覆盖数据分区的开始部分。
3. 数据分区的大小（以兆字节为单位）。
4. 挂载单元的名称必须与 **Where=** 指令中指定的目录匹配。例如，对于挂载在 **/var/lib/containers** 上的文件系统，该单元必须命名为 **var-lib-containers.mount**。
5. 对于用于容器存储的文件系统，必须启用 **prjquota** 挂载选项。

注意

在创建独立 **/var** 分区时，如果不同的实例类型没有相同的设备名称，则无法将不同的实例类型用于 worker 节点。

4. 再次运行 **openshift-install**，从 **manifest** 和 **openshift** 子目录中的一组文件创建 Ignition 配置：

```
$ openshift-install create ignition-configs --dir $HOME/clusterconfig
$ ls $HOME/clusterconfig/
auth  bootstrap.ign  master.ign  metadata.json  worker.ign
```

现在，您可以使用 Ignition 配置文件作为 vSphere 安装程序的输入来安装 Red Hat Enterprise Linux CoreOS（RHCOS）系统。

11.7.15. 创建集群

要创建 OpenShift Container Platform 集群，请等待您通过安装程序生成的 Ignition 配置文件所置备的机器上完成 bootstrap 过程。

先决条件

- 为集群创建所需的基础架构。
- 已获得安装程序并为集群生成了 Ignition 配置文件。
- 已使用 Ignition 配置文件为集群创建 RHCOS 机器。

流程

1. 监控 bootstrap 过程：

```
$ ./openshift-install --dir <installation_directory> wait-for bootstrap-complete \ 1
    --log-level=info 2
```

1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

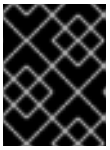
2 要查看不同的安装详情，请指定 **warn**、**debug** 或 **error**，而不要指定 **info**。

输出示例

```
INFO Waiting up to 30m0s for the Kubernetes API at https://api.test.example.com:6443...
INFO API v1.19.0 up
INFO Waiting up to 30m0s for bootstrapping to complete...
INFO It is now safe to remove the bootstrap resources
```

Kubernetes API 服务器提示已在 control plane 机器上完成 bootstrap 时，命令运行成功。

2. bootstrap 过程完成后，请从负载均衡器中删除 bootstrap 机器。



重要

此时您必须从负载均衡器中删除 bootstrap 机器。您还可以删除或重新格式化机器本身。

11.7.16. 使用 CLI 登录到集群

您可以通过导出集群 **kubeconfig** 文件，以默认系统用户身份登录集群。**kubeconfig** 文件包含关于集群的信息，供 CLI 用于将客户端连接到正确集群和 API 服务器。该文件特只适用于一个特定的集群，在 OpenShift Container Platform 安装过程中创建。

先决条件

- 已部署了 OpenShift Container Platform 集群。
- 已安装 **oc** CLI。

流程

1. 导出 **kubeadmin** 凭证：

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig
```
(1)

(1) 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

2. 使用导出的配置，验证能否成功运行 **oc** 命令：

```
$ oc whoami
```

输出示例

```
system:admin
```

11.7.17. 批准机器的证书签名请求

将机器添加到集群时，会为您添加的每台机器生成两个待处理证书签名请求（CSR）。您必须确认这些 CSR 已获得批准，或根据需要自行批准。客户端请求必须首先被批准，然后是服务器请求。

先决条件

- 您已将机器添加到集群中。

流程

1. 确认集群可以识别这些机器：

```
$ oc get nodes
```

输出示例

```
NAME     STATUS   ROLES   AGE  VERSION
master-0 Ready    master  63m  v1.19.0
master-1 Ready    master  63m  v1.19.0
master-2 Ready    master  64m  v1.19.0
```

输出将列出您创建的所有机器。

注意

在一些 CSR 被批准前，以上输出可能不包括计算节点（也称为 worker 节点）。

2. 检查待处理的 CSR，并确保可以看到添加到集群中的每台机器都有 **Pending** 或 **Approved** 状态的客户端请求：

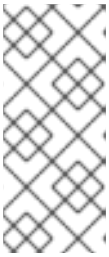
```
$ oc get csr
```

输出示例

```
NAME        AGE    REQUESTOR                                                   CONDITION
csr-8b2br   15m    system:serviceaccount:openshift-machine-config-operator:node-bootstrapper   Pending
csr-8vnps   15m    system:serviceaccount:openshift-machine-config-operator:node-bootstrapper   Pending
...
```

在本例中，两台机器加入了集群。您可能会在列表中看到更多已批准的 CSR。

3. 如果 CSR 没有获得批准，请在所添加机器的所有待处理 CSR 都处于 **Pending** 状态后，为您的集群机器批准这些 CSR：



注意

由于 CSR 会自动轮转，因此请在将机器添加到集群后一小时内批准您的 CSR。如果没有在一小时内批准，证书将会轮转，每个节点将会存在多个证书。您必须批准所有这些证书。批准客户端 CSR 后， Kubelet 为服务证书创建辅助 CSR，这需要手动批准。然后，如果 Kubelet 请求具有相同参数的新证书，则 **machine-approver** 会自动批准后续服务证书续订请求。

注意

对于在未启用机器 API 的平台中运行的集群，如裸机和其他用户置备的基础架构，必须采用一种方法自动批准 kubelet 提供证书请求（CSR）。如果没有批准请求，则 **oc exec**、**oc rsh** 和 **oc logs** 命令将无法成功，因为 API 服务器连接到 kubelet 时需要服务证书。与 Kubelet 端点联系的任何操作都需要此证书批准。这个方法必须监视新的 CSR，确认 CSR 由 **system:node** 或 **system:admin** 组中的 **node-bootstrapper** 服务帐户提交，并确认节点的身份。

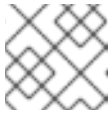
- 若要单独批准，请对每个有效的 CSR 运行以下命令：

```
$ oc adm certificate approve <csr_name> 1
```

1 **<csr_name>** 是当前 CSR 列表中 CSR 的名称。

- 要批准所有待处理的 CSR，请运行以下命令：

```
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}{{"\n"}}{{end}}{{end}}' | xargs --no-run-if-empty oc adm certificate approve
```



注意

在有些 CSR 被批准前，一些 Operator 可能无法使用。

4. 现在，您的客户端请求已被批准，您必须查看添加到集群中的每台机器的服务器请求：

```
$ oc get csr
```

输出示例

```
NAME        AGE     REQUESTOR                                                   CONDITION
csr-bfd72   5m26s   system:node:ip-10-0-50-126.us-east-2.compute.internal       Pending
csr-c57lv   5m26s   system:node:ip-10-0-95-157.us-east-2.compute.internal       Pending
...
```

5. 如果剩余的 CSR 没有被批准，且处于 **Pending** 状态，请批准集群机器的 CSR：

- 若要单独批准，请对每个有效的 CSR 运行以下命令：

```
$ oc adm certificate approve <csr_name> 1
```

1 **<csr_name>** 是当前 CSR 列表中 CSR 的名称。

- 要批准所有待处理的 CSR，请运行以下命令：

```
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}{{"\n"}}
{{end}}{{end}}' | xargs oc adm certificate approve
```

6. 批准所有客户端和服务器 CSR 后，器将处于 **Ready** 状态。运行以下命令验证：

```
$ oc get nodes
```

输出示例

```
NAME      STATUS    ROLES   AGE  VERSION
master-0  Ready     master  73m  v1.20.0
master-1  Ready     master  73m  v1.20.0
master-2  Ready     master  74m  v1.20.0
worker-0  Ready     worker  11m  v1.20.0
worker-1  Ready     worker  11m  v1.20.0
```

注意

批准服务器 CSR 后可能需要几分钟时间让机器转换为 **Ready** 状态。

其他信息

- 如需有关 CSR 的更多信息，请参阅证书签名请求。

11.7.18. 初始 Operator 配置

在 control plane 初始化后，您必须立即配置一些 Operator 以便它们都可用。

先决条件

- 您的 control plane 已初始化。

流程

1. 观察集群组件上线：

```
$ watch -n5 oc get clusteroperators
```

输出示例

```
NAME                                       VERSION AVAILABLE   PROGRESSING   DEGRADED
SINCE
authentication                             4.6.0   True        False         False      3h56m
cloud-credential                           4.6.0   True        False         False      29h
cluster-autoscaler                         4.6.0   True        False         False      29h
config-operator                            4.6.0   True        False         False      6h39m
console                                    4.6.0   True        False         False      3h59m
csi-snapshot-controller                    4.6.0   True        False         False      4h12m
dns                                        4.6.0   True        False         False      4h15m
etcd                                       4.6.0   True        False         False      29h
image-registry                             4.6.0   True        False         False      3h59m
ingress                                    4.6.0   True        False         False      4h30m
insights                                   4.6.0   True        False         False      29h
kube-apiserver                             4.6.0   True        False         False      29h
kube-controller-manager                    4.6.0   True        False         False      29h
kube-scheduler                             4.6.0   True        False         False      29h
kube-storage-version-migrator              4.6.0   True        False         False      4h2m
machine-api                                4.6.0   True        False         False      29h
machine-approver                           4.6.0   True        False         False      6h34m
machine-config                             4.6.0   True        False         False      3h56m
marketplace                                4.6.0   True        False         False      4h2m
monitoring                                 4.6.0   True        False         False      6h31m
network                                    4.6.0   True        False         False      29h
node-tuning                                4.6.0   True        False         False      4h30m
openshift-apiserver                        4.6.0   True        False         False      3h56m
openshift-controller-manager               4.6.0   True        False         False      4h36m
openshift-samples                          4.6.0   True        False         False      4h30m
operator-lifecycle-manager                 4.6.0   True        False         False      29h
operator-lifecycle-manager-catalog         4.6.0   True        False         False      29h
operator-lifecycle-manager-packageserver   4.6.0   True        False         False      3h59m
service-ca                                 4.6.0   True        False         False      29h
storage                                    4.6.0   True        False         False      4h30m
```

2. 配置不可用的 Operator。

11.7.18.1. 禁用默认的 OperatorHub 源

在 OpenShift Container Platform 安装过程中，默认为 OperatorHub 配置由红帽和社区项目提供的源内容的 operator 目录。在受限网络环境中，必须以集群管理员身份禁用默认目录。

流程

- 通过在 **OperatorHub** 对象中添加 **disableAllDefaultSources: true** 来禁用默认目录的源：

```
$ oc patch OperatorHub cluster --type json \
    -p '[{"op": "add", "path": "/spec/disableAllDefaultSources", "value": true}]'
```

提示

或者，您可以使用 Web 控制台管理目录源。在 **Administration → Cluster Settings → Global Configuration → OperatorHub** 页面中，点 **Sources** 选项卡，其中可创建、删除、禁用和启用单独的源。

11.7.18.2. 镜像 registry 存储配置

对于不提供默认存储的平台，Image Registry Operator 最初将不可用。安装后，您必须配置 registry 使用的存储，这样 Registry Operator 才可用。

示配置生产集群所需的持久性卷的说明。如果适用，显示有关将空目录配置为存储位置的说明，该位置只可用于非生产集群。

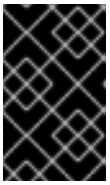
另外还提供了在升级过程中使用 **Recreate** rollout 策略来允许镜像 registry 使用块存储类型的说明。

11.7.18.2.1. 为 VMware vSphere 配置 registry 存储

作为集群管理员，在安装后需要配置 registry 来使用存储。

先决条件

- 具有 Cluster Administrator 权限
- VMware vSphere上有一个集群。
- 为集群置备的持久性存储，如 Red Hat OpenShift Container Storage。



重要

如果您只有一个副本，OpenShift Container Platform 支持对镜像 registry 存储的 **ReadWriteOnce** 访问。要部署支持高可用性的、带有两个或多个副本的镜像 registry，需要 **ReadWriteMany** 访问设置。

- 必须有“ 100Gi”容量。

重要

测试显示，在 RHEL 中使用 NFS 服务器作为核心服务的存储后端可能会出现问题。这包括 OpenShift Container Registry 和 Quay，Prometheus 用于监控存储，以及 Elasticsearch 用于日志存储。因此，不推荐使用 RHEL NFS 作为 PV 后端用于核心服务。

市场上的其他 NFS 实现可能没有这些问题。如需了解更多与此问题相关的信息，请联络相关的 NFS 厂商。

流程

1. 为了配置 registry 使用存储，需要修改 **configs.imageregistry/cluster** 资源中的 **spec.storage.pvc** 。

注意

使用共享存储时，请查看您的安全设置以防止被外部访问。

2. 验证您没有 registry pod:

```
$ oc get pod -n openshift-image-registry
```

注意

如果存储类型为 **emptyDIR**，则副本数不能超过 **1**。

3. 检查 registry 配置：

```
$ oc edit configs.imageregistry.operator.openshift.io
```

输出示例

```
storage:
  pvc:
    claim: 1
```

1 将 **claim** 字段留空以允许自动创建一个 **image-registry-storage** PVC。

4. 检查 **clusteroperator** 的状态：

```
$ oc get clusteroperator image-registry
```

11.7.18.2.2. 在非生产集群中配置镜像 registry 存储

您必须为 Image Registry Operator 配置存储。对于非生产集群，您可以将镜像 registry 设置为空目录。如果您这样做，重启 registry 后会丢失所有镜像。

流程

- 将镜像 registry 存储设置为空目录：

```
$ oc patch configs.imageregistry.operator.openshift.io cluster --type merge --patch '{"spec":{"storage":{"emptyDir":{}}}}'
```

警告

仅可为非生产集群配置这个选项。

如果在 Image Registry Operator 初始化其组件前运行此命令，**oc patch** 命令会失败并显示以下错误：

```
Error from server (NotFound): configs.imageregistry.operator.openshift.io "cluster" not found
```

等待几分钟，然后再次运行该命令。

11.7.18.2.3. 为 VMware vSphere 配置块 registry 存储

在作为集群管理员升级时，要允许镜像 registry 使用块存储类型，如 vSphere Virtual Machine Disk（VMDK），您可以使用 **Recreate** rollout 策略。

重要

支持块存储卷，但不建议将其用于生产环境中的镜像 registry。在块存储上配置 registry 的安装不具有高可用性，因为 registry 无法拥有多个副本。

流程

1. 要将镜像 registry 存储设置为块存储类型，对 registry 进行补丁，使其使用 **Recreate** rollout 策略，且仅使用 **1** 个副本运行：

```
$ oc patch config.imageregistry.operator.openshift.io/cluster --type=merge -p '{"spec":{"rolloutStrategy":"Recreate","replicas":1}}'
```

2. 为块存储设备置备 PV，并为该卷创建 PVC。请求的块卷使用 ReadWriteOnce（RWO）访问模式。

   a. 创建包含以下内容的 **pvc.yaml** 文件以定义 VMware vSphere **PersistentVolumeClaim**：

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: image-registry-storage 1
  namespace: openshift-image-registry 2
spec:
  accessModes:
  - ReadWriteOnce 3
  resources:
    requests:
      storage: 100Gi 4
```

   1. 代表 **PersistentVolumeClaim** 对象的唯一名称。
   2. **PersistentVolumeClaim** 对象的命名空间，即 **openshift-image-registry**。

3 持久性卷声明的访问模式。使用 **ReadWriteOnce** 时，单个节点可以通过读写权限挂载这个卷。

4 持久性卷声明的大小。

b. 从文件创建 **PersistentVolumeClaim** 对象：

```
$ oc create -f pvc.yaml -n openshift-image-registry
```

3. 编辑 registry 配置，使其可以正确引用 PVC:

```
$ oc edit config.imageregistry.operator.openshift.io -o yaml
```

输出示例

```
storage:
  pvc:
    claim: 1
```

1 通过创建自定义 PVC，您可以将 **claim** 字段留空以用于默认自动创建 **image-registry-storage** PVC。

有关配置 registry 存储以便引用正确的 PVC 的说明，请参阅为 vSphere 配置 registry。

11.7.19. 在用户置备的基础架构上完成安装

完成 Operator 配置后，可以在您提供的基础架构上完成集群安装。

先决条件

- 您的 control plane 已初始化。
- 已完成初始 Operator 配置。

流程

1. 使用以下命令确认所有集群组件都已在线：

```
$ watch -n5 oc get clusteroperators
```

输出示例

```
NAME                       VERSION AVAILABLE   PROGRESSING   DEGRADED
SINCE
authentication             4.6.0   True        False         False     3h56m
cloud-credential           4.6.0   True        False         False     29h
cluster-autoscaler         4.6.0   True        False         False     29h
config-operator            4.6.0   True        False         False     6h39m
console                    4.6.0   True        False         False     3h59m
csi-snapshot-controller    4.6.0   True        False         False     4h12m
dns                        4.6.0   True        False         False     4h15m
```

```
etcd                                       4.6.0   True        False         False      29h
image-registry                             4.6.0   True        False         False      3h59m
ingress                                    4.6.0   True        False         False      4h30m
insights                                   4.6.0   True        False         False      29h
kube-apiserver                             4.6.0   True        False         False      29h
kube-controller-manager                    4.6.0   True        False         False      29h
kube-scheduler                             4.6.0   True        False         False      29h
kube-storage-version-migrator              4.6.0   True        False         False      4h2m
machine-api                                4.6.0   True        False         False      29h
machine-approver                           4.6.0   True        False         False      6h34m
machine-config                             4.6.0   True        False         False      3h56m
marketplace                                4.6.0   True        False         False      4h2m
monitoring                                 4.6.0   True        False         False      6h31m
network                                    4.6.0   True        False         False      29h
node-tuning                                4.6.0   True        False         False      4h30m
openshift-apiserver                        4.6.0   True        False         False      3h56m
openshift-controller-manager               4.6.0   True        False         False      4h36m
openshift-samples                          4.6.0   True        False         False      4h30m
operator-lifecycle-manager                 4.6.0   True        False         False      29h
operator-lifecycle-manager-catalog         4.6.0   True        False         False      29h
operator-lifecycle-manager-packageserver   4.6.0   True        False         False      3h59m
service-ca                                 4.6.0   True        False         False      29h
storage                                    4.6.0   True        False         False      4h30m
```

或者，通过以下命令，如果所有集群都可用您会接到通知。它还检索并显示凭证：

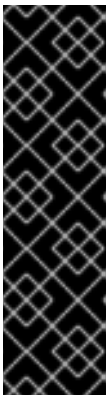
```
$ ./openshift-install --dir <installation_directory> wait-for install-complete 1
```

1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

输出示例

```
INFO Waiting up to 30m0s for the cluster to initialize...
```

Cluster Version Operator 完成从 Kubernetes API 服务器部署 OpenShift Container Platform 集群时，命令运行成功。



重要

- 安装程序生成的 Ignition 配置文件包含在 24 小时后过期的证书，然后在过期时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外情况是，您需要手动批准待处理的 **node-bootstrapper** 证书签名请求（CSR）来恢复 kubelet 证书。如需更多信息，请参阅*从过期的 control plane 证书中恢复*的文档。
- 建议您在生成 12 小时后使用 Ignition 配置文件，因为集群安装后 24 小时证书从 16 小时轮转至 22 小时。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中运行证书更新时避免安装失败。

2. 确认 Kubernetes API 服务器正在与 pod 通信。

   a. 要查看所有 pod 的列表，请使用以下命令：

```
$ oc get pods --all-namespaces
```

输出示例

```
NAMESPACE                      NAME                                        READY   STATUS
RESTARTS   AGE
openshift-apiserver-operator   openshift-apiserver-operator-85cb746d55-zqhs8   1/1
Running   1          9m
openshift-apiserver            apiserver-67b9g                             1/1     Running   0
3m
openshift-apiserver            apiserver-ljcmx                             1/1     Running   0
1m
openshift-apiserver            apiserver-z25h4                             1/1     Running   0
2m
openshift-authentication-operator authentication-operator-69d5d8bf84-vh2n8        1/1
Running   0          5m
...
```

b. 使用以下命令，查看上一命令的输出中所列 pod 的日志：

```
$ oc logs <pod_name> -n <namespace>
```
(1)

(1) 指定 pod 名称和命名空间，如上一命令的输出中所示。

如果 pod 日志显示，Kubernetes API 服务器可以与集群机器通信。

3. 在 Cluster registration 页面注册您的集群。

您可以按照将计算机器添加到 vSphere 的内容，在集群安装完成后添加额外的计算机器。

11.7.20. 备份 VMware vSphere 卷

OpenShift Container Platform 将新卷作为独立持久性磁盘置备，以便在集群中的任何节点上自由附加和分离卷。因此，无法备份使用快照的卷，也无法从快照中恢复卷。如需更多信息，请参阅 快照限制。

流程

要创建持久性卷的备份：

1. 停止使用持久性卷的应用程序。
2. 克隆持久性卷。
3. 重启应用程序。
4. 创建克隆的卷的备份。
5. 删除克隆的卷。

11.7.21. OpenShift Container Platform 的 Telemetry 访问

在 OpenShift Container Platform 4.6 中，默认运行的 Telemetry 服务提供有关集群健康状况和成功更新的指标，需要访问互联网。如果您的集群连接到互联网，Telemetry 会自动运行，而且集群会注册到 OpenShift Cluster Manager。

确认 OpenShift Cluster Manager 清单正确后，可以由 Telemetry 自动维护，也可以使用 OpenShift Cluster Manager 手动维护，使用订阅监控来跟踪帐户或多集群级别的 OpenShift Container Platform 订阅。

其他资源

- 有关 Telemetry 服务的更多信息，请参阅关于远程健康监控。

11.7.22. 后续步骤

- 自定义集群。
- 如果您用来安装集群的镜像 registry 具有一个可信任的 CA，通过配置额外的信任存储将其添加到集群中。
- 如果需要，您可以选择不使用远程健康报告。

11.8. 卸载在使用安装程序置备的基础架构的 VSPHERE 上的集群

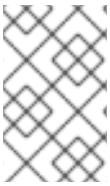
您可以使用安装程序置备的基础架构删除您在 VMware vSphere 实例中部署的集群。

注意

运行 **openshift-install destroy cluster** 命令时，卸载 OpenShift Container Platform 时，vSphere 卷不会被自动删除。集群管理员必须手动找到 vSphere 卷并删除它们。

11.8.1. 删除使用安装程序置备的基础架构的集群

您可以从云中删除使用安装程序置备的基础架构的集群。



注意

卸载后，检查云供应商是否有没有被正确移除的资源，特别是 User Provisioned Infrastructure（UPI）集群。可能存在安装程序没有创建的资源，或者安装程序无法访问的资源。

先决条件

- 有部署集群时所用的安装程序副本。
- 有创建集群时安装程序所生成的文件。

流程

1. 在用来安装集群的计算机中包含安装程序的目录中，运行以下命令：

```
$ ./openshift-install destroy cluster \
--dir <installation_directory> --log-level info 1 2
```

1. 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。
2. 要查看不同的详情，请指定 **warn**、**debug** 或 **error**，而不要指定 **info**。

注意

您必须为集群指定包含集群定义文件的目录。安装程序需要此目录中的 **metadata.json** 文件来删除集群。

2. 可选：删除 **<installation_directory>** 目录和 OpenShift Container Platform 安装程序。

第 12 章 在 VMC 上安装

12.1. 在 VMC 上安装集群

在 OpenShift Container Platform 版本 4.6 中，您可以通过将其部署到 VMware Cloud（VMC）on AWS 来在 VMware vSphere 上安装集群。

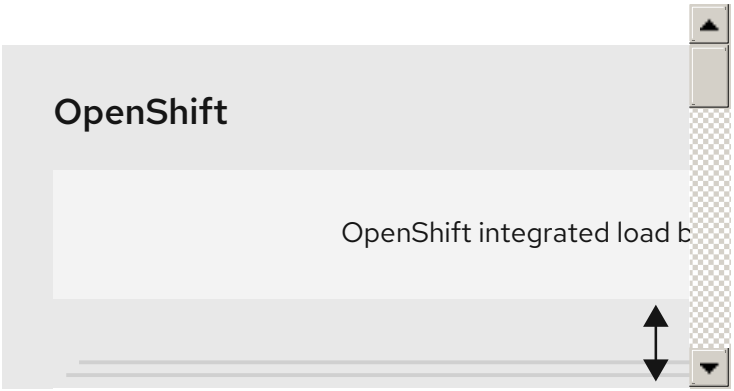
为 OpenShift Container Platform 部署配置了 VMC 环境后，您要使用堡垒管理主机中的 OpenShift Container Platform 安装程序，并位于 VMC 环境中。安装程序和 control plane 可以自动部署和管理 OpenShift Container Platform 集群所需的资源。

注意

OpenShift Container Platform 支持将集群部署到单个 VMware vCenter 中。不支持在多个 vCenter 上使用机器/机器集部署集群。

12.1.1. 为 vSphere 设置 VMC

您可以在 AWS 托管的 vSphere 集群上安装 OpenShift Container Platform（VMC），以便启用在混合云内部和内部管理应用程序。



在 VMware vSphere 上安装 OpenShift Container Platform 之前，您必须在 VMC 环境中配置多个选项。确定您的 VMC 环境有以下先决条件：

- 创建非专用的、启用了 DHCP 的、NSX-T 网络片段和子网。其他虚拟机（VM）可以托管在子网上，但至少有 8 个 IP 地址可用于 OpenShift Container Platform 部署。
- 在 DHCP 范围之外分配两个 IP 地址，并使用反向 DNS 记录配置它们。
  - **api.<cluster_name>.<base_domain>** 的 DNS 记录，指向分配的 IP 地址。
  - ***.apps.<cluster_name>.<base_domain>** 的 DNS 记录，指向分配的 IP 地址。
- 配置以下防火墙规则：
  - OpenShift Container Platform 计算网络和互联网间的 ANY:ANY 防火墙规则。节点和应用程序用于下载容器镜像。
  - 在安装主机和软件定义的数据中心（SDDC）管理网络之间的端口 443 的 ANY:ANY 防火墙规则。这可让您在部署过程中上传 Red Hat Enterprise Linux CoreOS（RHCOS）OVA。

- OpenShift Container Platform 计算网络和 vCenter 间的 HTTPS 防火墙规则。此连接允许 OpenShift Container Platform 与 vCenter 通信，以置备和管理节点、持久性卷声明（PVC）和其他资源。

- 您必须具有以下信息才能部署 OpenShift Container Platform:
  - OpenShift Container Platform 集群名称，如 **vmc-prod-1**。
  - 基本 DNS 名称，如 **companyname.com**。
  - 如果没有使用默认设置，则必须识别 Pod 网络 CIDR 和服务网络 CIDR，默认设置为 **10.128.0.0/14** 和 **172.30.0.0/16**。这些 CIDR 用于 pod 到 pod 和 pod 到服务的通信，且无法在外部访问，但不得与机构中的现有子网重叠。
  - 以下 vCenter 信息：
    - vCenter 主机名、用户名和密码
    - 数据中心名称，如 **SDDC-Datacenter**
    - 集群名称，如 **Cluster-1**
    - 网络名称
    - 数据存储名称，如 **WorkloadDatastore**

注意

建议在集群安装完成后将 vSphere 集群移到 VMC **Compute-ResourcePool** 资源池。

- 基于 Linux 的主机作为堡垒部署到 VMC。
  - 堡垒主机可以是 Red Hat Enterprise Linux（RHEL）或其他基于 Linux 的主机，它必须具有互联网连接，并可以将 OVA 上传到 ESXi 主机。
  - 将 OpenShift CLI 工具下载并安装到堡垒主机。
    - **openshift-install** 安装程序
    - OpenShift CLI（**oc**）工具

注意

您不能将 VMware NSX Container Plugin 用于 Kubernetes（NCP），NCP 不使用 NSX 作为 OpenShift SDN。目前 VMC 提供的 NSX 版本与 OpenShift Container Platform 认证的 NCP 版本不兼容。

但是，NSX DHCP 服务会通过全堆栈自动 OpenShift Container Platform 部署来管理虚拟机 IP 管理，节点可以手动或自动置备，可通过 Machine API 与 vSphere 集成。另外，还创建 NSX 防火墙规则，以便启用 OpenShift Container Platform 集群以及堡垒主机和 VMC vSphere 主机间的访问。

12.1.1.1. VMC Sizer 工具

AWS 上的 VMware Cloud 基于 AWS 裸机基础架构构建，这与运行 AWS 原生服务的裸机基础架构相同。

当在 AWS 软件定义的数据中心（SDDC）上部署 VMware 云时，您要将这些物理服务器节点以单一租户方式运行 VMware ESXi hypervisor。这意味着其他使用 VMC 的用户无法访问物理基础结构。务必要考虑托管您的虚拟基础架构所需的物理主机数量。

要确定这一点，VMware 在 AWS Sizer 上提供 VMC。使用这个工具，您可以定义要在 VMC 上托管的资源：

- 工作负载类型
- 虚拟机总数
- 规格信息，例如：
  - 存储要求
  - vCPUs
  - vRAM
  - 过量使用比例

通过这些信息，sizer 工具可以根据 VMware 最佳实践生成报告，并推荐集群配置和您需要的主机数量。

12.1.2. vSphere 先决条件

- 置备块 registry 存储。如需有关持久性存储的更多信息，请参阅了解持久性存储。
- 查看有关 OpenShift Container Platform 安装和更新流程的详细信息。
- 如果使用防火墙，则必须将其配置为允许集群需要访问的站点。

注意

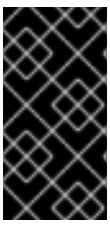
如果您要配置代理，请务必也要查看此站点列表。

12.1.3. OpenShift Container Platform 的互联网访问

在 OpenShift Container Platform 4.6 中，您需要访问互联网来安装集群。

您必须具有以下互联网访问权限：

- 访问 OpenShift Cluster Manager 以下载安装程序并执行订阅管理。如果集群可以访问互联网，并且没有禁用 Telemetry，该服务会自动授权您的集群。
- 访问 Quay.io，以获取安装集群所需的软件包。
- 获取执行集群更新所需的软件包。



重要

如果您的集群无法直接访问互联网，则可以在置备的某些类基础架构上执行受限网络安装。在此过程中，您要下载所需的内容，并使用它在镜像 registry（mirror registry） 中填充安装集群并生成安装程序所需的软件包。对于某些安装类型，集群要安装到的环境不需要访问互联网。在更新集群之前，要更新 registry 镜像系统中的内容。

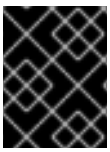
12.1.4. VMware vSphere 基础架构要求

您必须在满足您使用的组件要求的 VMware vSphere 版本 6 或 7 实例上安装 OpenShift Container Platform 集群。

表 12.1. VMware 组件支持的最低 vSphere 版本

组件	最低支持版本	描述
虚拟机监控程序	vSphere 6.5 及之后的版本 13	此版本是 Red Hat Enterprise Linux CoreOS(RHCOS)支持的最低版本。请查看 Red Hat Enterprise Linux 8 支持的管理程序列表。
使用 in-tree 驱动程序存储	vSphere 6.5 及之后的版本	此插件使用 OpenShift Container Platform 中包含的 vSphere 的树内存储驱动程序创建 vSphere 存储。

如果您使用 vSphere 版本 6.5 实例，请在安装 OpenShift Container Platform 前考虑升级到 6.7U3 或 7.0。



重要

您必须确保在安装 OpenShift Container Platform 前同步 ESXi 主机上的时间。请参阅 VMware 文档中的编辑主机时间配置。

12.1.5. 网络连接要求

您必须配置机器之间的网络连接，以允许 OpenShift Container Platform 集群组件进行通信。

查看有关所需网络端口的以下详细信息。

表 12.2. 用于全机器到所有机器通信的端口

协议	端口	描述
ICMP	N/A	网络可访问性测试
TCP	**1936**	指标
	9000-9999	主机级别的服务，包括端口 **9100-9101** 上的节点导出器和端口 **9099** 上的 Cluster Version Operator。
	10250-10259	Kubernetes 保留的默认端口
	10256	openshift-sdn
UDP	**4789**	虚拟可扩展 LAN(VXLAN)

协议	端口	描述
	6081	Geneve
	9000-9999	主机级别的服务，包括端口 **9100-9101** 上的节点导出器。
	500	IPsec IKE 数据包
	4500	IPsec NAT-T 数据包
TCP/UDP	**30000-32767**	Kubernetes 节点端口
ESP	N/A	IPsec Encapsulating Security Payload(ESP)

表 12.3. 用于所有机器控制平面通信的端口

协议	端口	描述
TCP	**6443**	Kubernetes API

表 12.4. control plane 机器用于 control plane 机器通信的端口

协议	端口	描述
TCP	**2379-2380**	etcd 服务器和对等端口

12.1.6. vCenter 要求

在使用安装程序置备的基础架构的 vCenter 上安装 OpenShift Container Platform 集群前，您必须准备自己的环境。

所需的 vCenter 帐户权限
要在 vCenter 中安装 OpenShift Container Platform 集群，安装程序需要一个具有特权的帐户来读取和创建所需资源。使用具有全局管理特权的帐户是访问所有必要权限的最简单方式。

如果无法使用具有全局管理特权的帐户，您必须创建角色来授予 OpenShift Container Platform 集群安装所需的权限。虽然大多数权限始终是必需的，但是一些权限只有在计划安装程序需要在您的 vCenter 实例中置备一个包含 OpenShift Container Platform 集群的文件夹时（这是默认行为）才需要。您必须为指定对象创建或修改 vSphere 角色，才能授予所需的权限。

如果安装程序创建 vSphere 虚拟机文件夹，则需要额外的角色。

例 12.1. 安装所需的角色和权限

角色的 vSphere 对象	何时需要	所需的权限

角色的 vSphere 对象	何时需要	所需的权限
vSphere vCenter	Always	**Cns.Searchable** **InventoryService.Tagging.AttachTag** **InventoryService.Tagging.CreateCategory** **InventoryService.Tagging.CreateTag** **InventoryService.Tagging.DeleteCategory** **InventoryService.Tagging.DeleteTag** **InventoryService.Tagging.EditCategory** **InventoryService.Tagging.EditTag** **Sessions.ValidateSession** **StorageProfile.View**
vSphere vCenter Cluster	Always	**Host.Config.Storage** **Resource.AssignVMToPool** **VApp.AssignResourcePool** **VApp.Import** **VirtualMachine.Config.AddNewDisk**
vSphere Datastore	Always	**Datastore.AllocateSpace** **Datastore.Browse** **Datastore.FileManagement**
vSphere 端口组	Always	**Network.Assign**

角色的 vSphere 对象	何时需要	所需的权限
虚拟机文件夹	Always	**Resource.AssignVMToPool VApp.Import VirtualMachine.Config.Add ExistingDisk VirtualMachine.Config.Add NewDisk VirtualMachine.Config.Add RemoveDevice VirtualMachine.Config.Adva ncedConfig VirtualMachine.Config.Anno tation VirtualMachine.Config.CPU Count VirtualMachine.Config.Disk Extend VirtualMachine.Config.Disk Lease VirtualMachine.Config.Edit Device VirtualMachine.Config.Mem ory VirtualMachine.Config.Rem oveDisk VirtualMachine.Config.Rena me VirtualMachine.Config.Rese tGuestInfo VirtualMachine.Config.Reso urce VirtualMachine.Config.Setti ngs VirtualMachine.Config.Upgr adeVirtualHardware VirtualMachine.Interact.Gue stControl VirtualMachine.Interact.Pow erOff VirtualMachine.Interact.Pow erOn VirtualMachine.Interact.Res et VirtualMachine.Inventory.Cr eate VirtualMachine.Inventory.Cr eateFromExisting VirtualMachine.Inventory.D elete VirtualMachine.Provisionin g.Clone**

角色的 vSphere 对象	何时需要	所需的权限
vSphere vCenter Datacenter	如果安装程序创建虚拟机文件夹	**Resource.AssignVMToPool** **VApp.Import** **VirtualMachine.Config.AddExistingDisk** **VirtualMachine.Config.AddNewDisk** **VirtualMachine.Config.AddRemoveDevice** **VirtualMachine.Config.AdvancedConfig** **VirtualMachine.Config.Annotation** **VirtualMachine.Config.CPUCount** **VirtualMachine.Config.DiskExtend** **VirtualMachine.Config.DiskLease** **VirtualMachine.Config.EditDevice** **VirtualMachine.Config.Memory** **VirtualMachine.Config.RemoveDisk** **VirtualMachine.Config.Rename** **VirtualMachine.Config.ResetGuestInfo** **VirtualMachine.Config.Resource** **VirtualMachine.Config.Settings** **VirtualMachine.Config.UpgradeVirtualHardware** **VirtualMachine.Interact.GuestControl** **VirtualMachine.Interact.PowerOff** **VirtualMachine.Interact.PowerOn** **VirtualMachine.Interact.Reset** **VirtualMachine.Inventory.Create** **VirtualMachine.Inventory.CreateFromExisting** **VirtualMachine.Inventory.Delete** **VirtualMachine.Provisioning.Clone** **Folder.Create** **Folder.Delete**

此外，用户需要一些 **ReadOnly** 权限，某些角色需要权限来提升对子对象的权限。这些设置会根据您是否将集群安装到现有文件夹而有所不同。

例 12.2. 所需的权限和传播设置

vSphere 对象	文件夹类型	传播到子对象	所需的权限
vSphere vCenter	Always	False	列出所需的权限
vSphere vCenter Datacenter	现有文件夹	False	**ReadOnly** 权限
	安装程序创建文件夹	True	列出所需的权限
vSphere vCenter Cluster	Always	True	列出所需的权限
vSphere vCenter Datastore	Always	False	列出所需的权限
vSphere Switch	Always	False	**ReadOnly** 权限
vSphere 端口组	Always	False	列出所需的权限
vSphere vCenter Virtual Machine Folder	现有文件夹	True	列出所需的权限

有关只使用所需权限创建帐户的更多信息，请参阅 vSphere 文档中的 vSphere 权限和用户管理任务。

将 OpenShift Container Platform 与 vMotion 搭配使用

如果要在 vSphere 环境中使用 vMotion，请在安装 OpenShift Container Platform 集群前考虑以下内容。

- OpenShift Container Platform 通常支持仅用于计算的 vMotion。使用 Storage vMotion 可能会导致问题且不被支持。
  为了帮助确保计算和 control plane 节点的正常运行时间，建议您遵循 VMware 最佳实践进行 vMotion。还建议使用 VMware 反关联性规则来改进 OpenShift Container Platform 在维护或硬件问题期间的可用性。

  有关 vMotion 和 anti-affinity 规则的更多信息，请参阅 VMware vSphere 文档 了解 vMotion 网络要求和虚拟机反关联性规则。

- 如果您在 pod 中使用 vSphere 卷，请手动或通过 Storage vMotion 在数据存储间迁移虚拟机，从而导致 OpenShift Container Platform 持久性卷(PV)对象中的无效引用。这些引用可防止受影响的 pod 启动，并可能导致数据丢失。

- 同样，OpenShift Container Platform 不支持在数据存储间有选择地迁移 VMDK、使用数据存储集群进行虚拟机置备、动态或静态置备 PV，或使用作为数据存储集群一部分的数据存储进行 PV 的动态或静态置备。

集群资源
当部署使用安装程序置备的基础架构的 OpenShift Container Platform 集群时，安装程序必须能够在 vCenter 实例中创建多个资源。

标准 OpenShift Container Platform 安装会创建以下 vCenter 资源：

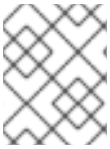
- 1 个文件夹
- 1 标签（Tag）类别
- 1 个标签（Tag）
- 虚拟机:
  - 1 个模板
  - 1 个临时 bootstrap 节点
  - 3 个 control plane 节点
  - 3 个计算机器

虽然这些资源使用了 856 GB 存储，但 bootstrap 节点会在集群安装过程中被销毁。使用标准集群至少需要 800 GB 存储。

如果部署了更多计算机器，OpenShift Container Platform 集群将使用更多存储。

集群的限制
可用资源因集群而异。vCenter 中可能的集群数量主要受可用存储空间以及对所需资源数量的限制。确保考虑集群创建的 vCenter 资源的限制和部署集群所需的资源，如 IP 地址和网络。

网络要求
网络必须使用 DHCP，并确保 DHCP 服务器被配置为为集群机器提供持久的 IP 地址。



注意

在安装开始前，持久性 IP 地址不可用。分配 DHCP 范围后，在安装后使用持久 IP 地址手动替换分配。

另外，在安装 OpenShift Container Platform 集群前，必须创建以下网络资源：

注意

建议集群中的每个 OpenShift Container Platform 节点都可以访问可通过 DHCP 发现的网络时间协议（NTP）服务器。没有 NTP 服务器也可安装。但是，异步服务器时钟将导致错误，NTP 服务器会阻止。

所需的 IP 地址
安装程序置备的 vSphere 安装需要这些静态 IP 地址：

- API 地址用于访问集群 API。
- Ingress 地址用于集群入口流量。
- 当将集群从版本 4.5 升级到 4.6 时，会使用 control plane 节点地址。

安装 OpenShift Container Platform 集群时，必须向安装程序提供这些 IP 地址。

DNS 记录

您必须在正确的 DNS 服务器中为托管 OpenShift Container Platform 集群的 vCenter 实例创建两个静态 IP 地址的 DNS 记录。在每个记录中，**<cluster_name>** 是集群名称，**<base_domain>** 是您在安装集群时指定的集群基域。完整的 DNS 记录采用如下格式: **<component>.<cluster_name>.<base_domain>.**。

表 12.5. 所需的 DNS 记录

组件	记录	描述
API VIP	**api.<cluster_name>.<base_domain>.**	此 DNS A/AAAA 或 CNAME 记录必须指向 control plane 机器的负载均衡器。此记录必须能由集群外的客户端和集群内的所有节点解析。
Ingress VIP	***.apps.<cluster_name>.<base_domain>.**	通配符 DNS A/AAAA 或 CNAME 记录，指向以运行入口路由器 Pod 的机器（默认为 worker 节点）为目标的负载均衡器。此记录必须能由集群外的客户端和集群内的所有节点解析。

12.1.7. 生成 SSH 私钥并将其添加到代理中

如果要在集群上执行安装调试或灾难恢复，则必须为 **ssh-agent** 和安装程序提供 SSH 密钥。您可以使用此密钥访问公共集群中的 bootstrap 机器来排除安装问题。

注意

在生产环境中，您需要进行灾难恢复和调试。

您可以使用此密钥以 **core** 用户身份通过 SSH 连接到 master 节点。在部署集群时，此密钥会添加到 **core** 用户的 **~/.ssh/authorized_keys** 列表中。

注意

您必须使用一个本地密钥，而不要使用在特定平台上配置的密钥，如 AWS 密钥对。

流程

1. 如果还没有为计算机上免密码身份验证而配置的 SSH 密钥，请创建一个。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ ssh-keygen -t ed25519 -N '' \
  -f <path>/<file_name> 1
```

1 指定新 SSH 密钥的路径和文件名，如 **~/.ssh/id_rsa**。如果您已有密钥对，请确保您的公钥位于 **~/.ssh** 目录中。

运行此命令会在指定的位置生成不需要密码的 SSH 密钥。

注意

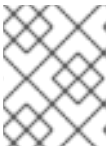
如果您计划在 **x86_64** 架构中安装使用 FIPS 验证的/Modules in Process 加密库的 OpenShift Container Platform 集群，不要创建使用 **ed25519** 算法的密钥。反之，创建一个使用 **rsa** 或 **ecdsa** 算法的密钥。

2. 作为后台任务启动 **ssh-agent** 进程：

```
$ eval "$(ssh-agent -s)"
```

输出示例

```
Agent pid 31874
```



注意

如果您的集群采用 FIPS 模式，则只使用 FIPS 兼容算法来生成 SSH 密钥。密钥必须是 RSA 或 ECDSA。

3. 将 SSH 私钥添加到 **ssh-agent**：

```
$ ssh-add <path>/<file_name> (1)
```

输出示例

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

(1) 指定 SSH 私钥的路径和文件名，如 **~/.ssh/id_rsa**

后续步骤

- 在安装 OpenShift Container Platform 时，为安装程序提供 SSH 公钥。

12.1.8. 获取安装程序

在安装 OpenShift Container Platform 之前，将安装文件下载到本地计算机上。

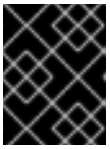
先决条件

- 运行 Linux 或 macOS 的计算机，本地磁盘空间为 500 MB

流程

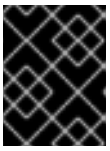
1. 访问 OpenShift Cluster Manager 站点的 Infrastructure Provider 页面。如果您有红帽帐号，请使用自己的凭证登录。如果没有，请创建一个帐户。
2. 选择您的基础架构供应商。

3. 进入适用于您的安装类型的页面，下载您的操作系统的安装程序，并将文件放在要保存安装配置文件的目录中。。



重要

安装程序会在用来安装集群的计算机上创建若干文件。在完成集群安装后，您必须保留安装程序和安装程序所创建的文件。这两个文件都需要删除集群。



重要

删除安装程序创建的文件不会删除您的集群，即使集群在安装过程中失败也是如此。要删除集群，为特定云供应商完成 OpenShift Container Platform 卸载流程。

4. 提取安装程序。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ tar xvf openshift-install-linux.tar.gz
```

5. 从 Red Hat OpenShift Cluster Manager 下载安装 pull secret 。通过此 pull secret，您可以进行所含授权机构提供的服务的身份验证，这些服务包括为 OpenShift Container Platform 组件提供容器镜像的 Quay.io。

12.1.9. 在您的系统信任中添加 vCenter root CA 证书

由于安装程序需要访问 vCenter 的 API，所以必须在安装 OpenShift Container Platform 集群前将 vCenter 的可信 root CA 证书添加到系统信任中。

流程

1. 在 vCenter 主页中下载 vCenter 的 root CA 证书。在 vSphere Web Services SDK 部分点击 **Download trusted root CA certificates**。**<vCenter>/certs/download.zip** 文件下载。

2. 提取包含 vCenter root CA 证书的压缩文件。压缩文件的内容类似以下文件结构：

```
certs
├── lin
│   ├── 108f4d17.0
│   ├── 108f4d17.r1
│   ├── 7e757f6a.0
│   ├── 8e4f8471.0
│   └── 8e4f8471.r0
├── mac
│   ├── 108f4d17.0
│   ├── 108f4d17.r1
│   ├── 7e757f6a.0
│   ├── 8e4f8471.0
│   └── 8e4f8471.r0
└── win
    ├── 108f4d17.0.crt
    ├── 108f4d17.r1.crl
    ├── 7e757f6a.0.crt
    ├── 8e4f8471.0.crt
    └── 8e4f8471.r0.crl

3 directories, 15 files
```

3. 将您的操作系统的文件添加到系统信任中。例如，在 Fedora 操作系统上运行以下命令：

```
# cp certs/lin/* /etc/pki/ca-trust/source/anchors
```

4. 更新您的系统信任关系。例如，在 Fedora 操作系统上运行以下命令：

```
# update-ca-trust extract
```

12.1.10. 部署集群

您可以在兼容云平台中安装 OpenShift Container Platform。

重要

安装程序的 **create cluster** 命令只能在初始安装过程中运行一次。

先决条件

- 配置托管集群的云平台的帐户。
- 获取 OpenShift Container Platform 安装程序以及集群的 pull secret。

流程

1. 更改为包含安装程序的目录并初始化集群部署：

```
$ ./openshift-install create cluster --dir <installation_directory> \ 1
    --log-level=info 2
```

1 对于 **<installation_directory>**，请指定用于保存安装程序所创建的文件的目录名称。

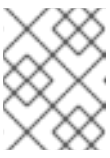
2 要查看不同的安装详情，请指定 **warn**、**debug** 或 **error**，而不要指定 **info**。

重要

指定一个空目录。一些安装信息，如 bootstrap X.509 证书，有较短的过期间隔，因此不要重复使用安装目录。如果要重复使用另一个集群安装中的个别文件，可以将其复制到您的目录中。但是，一些安装数据的文件名可能会在发行版本之间有所改变。从 OpenShift Container Platform 老版本中复制安装文件时要格外小心。

在提示符处提供值：

a. 可选：选择用来访问集群机器的 SSH 密钥。



注意

对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 **ssh-agent** 进程使用的 SSH 密钥。

b. 选择 **vsphere** 作为目标平台。

c. 指定 vCenter 实例的名称。

d. 指定创建集群所需的权限的 vCenter 帐户的用户名和密码。
安装程序连接到您的 vCenter 实例。

e. 选择要连接的 vCenter 实例中的数据中心。

f. 选择要使用的默认 vCenter 数据存储。

注意

数据存储和集群名称不能超过 60 个字符，因此请确保组合字符串长度不超过 60 个字符的限制。

g. 选择要在其中安装 vCenter 集群的 OpenShift Container Platform 集群。安装程序使用 vSphere 集群的 root 资源池作为默认资源池。

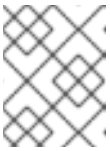
h. 选择包含您配置的虚拟 IP 地址和 DNS 记录的 vCenter 实例中的网络。

i. 输入您为 control plane API 访问配置的虚拟 IP 地址。

j. 输入您为集群入口配置的虚拟 IP 地址。

k. 输入基域。这个基域必须与您配置的 DNS 记录中使用的域相同。

l. 为集群输入一个描述性名称。集群名称必须与您配置的 DNS 记录中使用的相同。



注意

数据存储和集群名称不能超过 60 个字符，因此请确保组合字符串长度不超过 60 个字符的限制。

m. 粘贴 Red Hat OpenShift Cluster Manager 中的 pull secret 。

重要

使用托管在 VMC 环境中的堡垒中的 **openshift-install** 命令。

注意

如果您在主机上配置的云供应商帐户没有足够的权限来部署集群，安装过程将会停止，并且显示缺少的权限。

集群部署完成后，终端会显示访问集群的信息，包括指向其 Web 控制台的链接和 **kubeadmin** 用户的凭证。

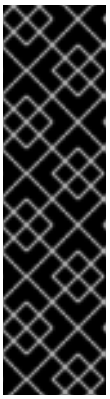
输出示例

```
...
INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run 'export
KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'
INFO Access the OpenShift web-console here: https://console-openshift-
console.apps.mycluster.example.com
```

```
INFO Login to the console with user: "kubeadmin", and password: "4vYBz-Ee6gm-ymBZj-Wt5AL"
INFO Time elapsed: 36m22s
```

注意

当安装成功时，集群访问和凭证信息还会输出到 **<installation_directory>/.openshift_install.log**。



重要

- 安装程序生成的 Ignition 配置文件包含在 24 小时后过期的证书，然后在过期时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外情况是，您需要手动批准待处理的 **node-bootstrapper** 证书签名请求（CSR）来恢复 kubelet 证书。如需更多信息，请参阅*从过期的 control plane 证书中恢复*的文档。
- 建议您在生成 12 小时后使用 Ignition 配置文件，因为集群安装后 24 小时证书从 16 小时轮转至 22 小时。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中运行证书更新时避免安装失败。

重要

您不得删除安装程序或安装程序所创建的文件。需要这两者才能删除集群。

12.1.11. 通过下载二进制文件安装 OpenShift CLI

您需要安装 CLI（**oc**） 来使用命令行界面与 OpenShift Container Platform 进行交互。您可在 Linux、Windows 或 macOS 上安装 **oc**。

重要

如果安装了旧版本的 **oc**，则无法使用 OpenShift Container Platform 4.6 中的所有命令。下载并安装新版本的 **oc**。

12.1.11.1. 在 Linux 上安装 OpenShift CLI

您可以按照以下流程在 Linux 上安装 OpenShift CLI（**oc**）二进制文件。

流程

1. 进入到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 在 **Version** 下拉菜单中选择相应的版本。
3. 单击 **OpenShift v4.6 Linux 客户端** 条目旁边的 **Download Now**，再保存文件。
4. 解包存档：

```
$ tar xvzf <file>
```

5. 把 **oc** 二进制代码放到 **PATH** 中的目录中。
   执行以下命令可以查看当前的 **PATH** 设置：

```
$ echo $PATH
```

安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
$ oc <command>
```

12.1.11.2. 在 Windows 上安装 OpenShift CLI

您可以按照以下流程在 Windows 上安装 OpenShift CLI（**oc**）二进制代码。

流程

1. 进入到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 在 **Version** 下拉菜单中选择相应的版本。
3. 单击 **OpenShift v4.6 Windows 客户端**条目旁边的 **Download Now**，再保存文件。
4. 使用 ZIP 程序解压存档。
5. 把 **oc** 二进制代码放到 **PATH** 中的目录中。
   要查看您的 **PATH**，请打开命令提示窗口并执行以下命令：

```
C:\> path
```

安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
C:\> oc <command>
```

12.1.11.3. 在 macOS 上安装 OpenShift CLI

您可以按照以下流程在 macOS 上安装 OpenShift CLI（**oc**）二进制代码。

流程

1. 进入到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 在 **Version** 下拉菜单中选择相应的版本。
3. 单击 **OpenShift v4.6 MacOSX 客户端**条目旁边的 **Download Now**，再保存文件。
4. 解包和解压存档。
5. 将 **oc** 二进制文件移到 PATH 的目录中。
   要查看您的 **PATH**，打开一个终端窗口并执行以下命令：

```
$ echo $PATH
```

安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
$ oc <command>
```

12.1.12. 使用 CLI 登录到集群

您可以通过导出集群 **kubeconfig** 文件，以默认系统用户身份登录集群。**kubeconfig** 文件包含关于集群的信息，供 CLI 用于将客户端连接到正确集群和 API 服务器。该文件特只适用于一个特定的集群，在 OpenShift Container Platform 安装过程中创建。

先决条件

- 已部署了 OpenShift Container Platform 集群。
- 已安装 **oc** CLI。

流程

1. 导出 **kubeadmin** 凭证：

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

2. 使用导出的配置，验证能否成功运行 **oc** 命令：

```
$ oc whoami
```

输出示例

```
system:admin
```

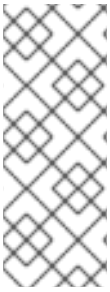
12.1.13. 创建 registry 存储

安装集群后，必须为 registry Operator 创建存储。

12.1.13.1. 安装过程中删除的镜像 registry

在不提供可共享对象存储的平台上，OpenShift Image Registry Operator bootstraps 本身的状态是 **Removed**。这允许 **openshift-installer** 在这些平台类型上完成安装。

将 **ManagementState** Image Registry Operator 配置从 **Removed** 改为 **Managed**。



注意

Prometheus 控制台提供了一个 **ImageRegistryRemoved** 警报，例如：

"Image Registry has been removed.**ImageStreamTags**, **BuildConfigs** and **DeploymentConfigs** which reference **ImageStreamTags** may not work as expected.Please configure storage and update the config to **Managed** state by editing configs.imageregistry.operator.openshift.io."

12.1.13.2. 镜像 registry 存储配置

对于不提供默认存储的平台，Image Registry Operator 最初将不可用。安装后，您必须配置 registry 使用的存储，这样 Registry Operator 才可用。

示配置生产集群所需的持久性卷的说明。如果适用，显示有关将空目录配置为存储位置的说明，该位置只可用于非生产集群。

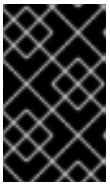
另外还提供了在升级过程中使用 **Recreate** rollout 策略来允许镜像 registry 使用块存储类型的说明。

12.1.13.2.1. 为 VMware vSphere 配置 registry 存储

作为集群管理员，在安装后需要配置 registry 来使用存储。

先决条件

- 具有 Cluster Administrator 权限
- VMware vSphere上有一个集群。
- 为集群置备的持久性存储，如 Red Hat OpenShift Container Storage。



重要

如果您只有一个副本，OpenShift Container Platform 支持对镜像 registry 存储的 **ReadWriteOnce** 访问。要部署支持高可用性的、带有两个或多个副本的镜像 registry，需要 **ReadWriteMany** 访问设置。

- 必须有“ 100Gi”容量。

重要

测试显示，在 RHEL 中使用 NFS 服务器作为核心服务的存储后端可能会出现问题。这包括 OpenShift Container Registry 和 Quay，Prometheus 用于监控存储，以及 Elasticsearch 用于日志存储。因此，不推荐使用 RHEL NFS 作为 PV 后端用于核心服务。

市场上的其他 NFS 实现可能没有这些问题。如需了解更多与此问题相关的信息，请联络相关的 NFS 厂商。

流程

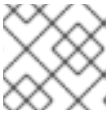
1. 为了配置 registry 使用存储，需要修改 **configs.imageregistry/cluster** 资源中的 **spec.storage.pvc** 。

注意

使用共享存储时，请查看您的安全设置以防止被外部访问。

2. 验证您没有 registry pod:

```
$ oc get pod -n openshift-image-registry
```



注意

如果存储类型为 **emptyDIR**，则副本数不能超过 **1**。

3. 检查 registry 配置：

```
$ oc edit configs.imageregistry.operator.openshift.io
```

输出示例

```
storage:
  pvc:
    claim: 1
```

1 将 **claim** 字段留空以允许自动创建一个 **image-registry-storage** PVC。

4. 检查 **clusteroperator** 的状态：

```
$ oc get clusteroperator image-registry
```

12.1.13.2.2. 为 VMware vSphere 配置块 registry 存储

在作为集群管理员升级时，要允许镜像 registry 使用块存储类型，如 vSphere Virtual Machine Disk（VMDK），您可以使用 **Recreate** rollout 策略。

重要

支持块存储卷，但不建议将其用于生产环境中的镜像 registry。在块存储上配置 registry 的安装不具有高可用性，因为 registry 无法拥有多个副本。

流程

1. 要将镜像 registry 存储设置为块存储类型，对 registry 进行补丁，使其使用 **Recreate** rollout 策略，且仅使用 **1** 个副本运行：

```
$ oc patch config.imageregistry.operator.openshift.io/cluster --type=merge -p '{"spec":
{"rolloutStrategy":"Recreate","replicas":1}}'
```

2. 为块存储设备置备 PV，并为该卷创建 PVC。请求的块卷使用 ReadWriteOnce（RWO）访问模式。

   a. 创建包含以下内容的 **pvc.yaml** 文件以定义 VMware vSphere **PersistentVolumeClaim**：

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: image-registry-storage 1
  namespace: openshift-image-registry 2
spec:
  accessModes:
  - ReadWriteOnce 3
  resources:
    requests:
      storage: 100Gi 4
```

1 代表 **PersistentVolumeClaim** 对象的唯一名称。

2 **PersistentVolumeClaim** 对象的命名空间，即 **openshift-image-registry**。

3 持久性卷声明的访问模式。使用 **ReadWriteOnce** 时，单个节点可以通过读写权限挂载这个卷。

4 持久性卷声明的大小。

b. 从文件创建 **PersistentVolumeClaim** 对象：

```
$ oc create -f pvc.yaml -n openshift-image-registry
```

3. 编辑 registry 配置，使其可以正确引用 PVC:

```
$ oc edit config.imageregistry.operator.openshift.io -o yaml
```

输出示例

```
storage:
  pvc:
    claim: 1
```

1 通过创建自定义 PVC，您可以将 **claim** 字段留空以用于默认自动创建 **image-registry-storage** PVC。

有关配置 registry 存储以便引用正确的 PVC 的说明，请参阅为 vSphere 配置 registry。

12.1.14. 备份 VMware vSphere 卷

OpenShift Container Platform 将新卷作为独立持久性磁盘置备，以便在集群中的任何节点上自由附加和分离卷。因此，无法备份使用快照的卷，也无法从快照中恢复卷。如需更多信息，请参阅 快照限制。

流程

要创建持久性卷的备份：

1. 停止使用持久性卷的应用程序。
2. 克隆持久性卷。
3. 重启应用程序。
4. 创建克隆的卷的备份。
5. 删除克隆的卷。

12.1.15. OpenShift Container Platform 的 Telemetry 访问

在 OpenShift Container Platform 4.6 中，默认运行的 Telemetry 服务提供有关集群健康状况和成功更新的指标，需要访问互联网。如果您的集群连接到互联网，Telemetry 会自动运行，而且集群会注册到 OpenShift Cluster Manager。

确认 OpenShift Cluster Manager 清单正确后，可以由 Telemetry 自动维护，也可以使用 OpenShift Cluster Manager 手动维护，使用订阅监控来跟踪帐户或多集群级别的 OpenShift Container Platform 订阅。

其他资源

- 有关 Telemetry 服务的更多信息，请参阅关于远程健康监控。

12.1.16. 后续步骤

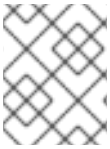
- 自定义集群。
- 如果需要，您可以选择不使用远程健康报告。
- 设置 registry 并配置 registry 存储。

12.2. 使用自定义在 VMC 上安装集群

在 OpenShift Container Platform 版本 4.6 中，您可以使用安装程序置备的基础架构在 VMware vSphere 实例上安装集群，方法是将其部署到 VMware Cloud (VMC) on AWS。

为 OpenShift Container Platform 部署配置了 VMC 环境后，您要使用堡垒管理主机中的 OpenShift Container Platform 安装程序，并位于 VMC 环境中。安装程序和 control plane 可以自动部署和管理 OpenShift Container Platform 集群所需的资源。

要自定义 OpenShift Container Platform 安装，请在安装集群前修改 **install-config.yaml** 文件中的参数。

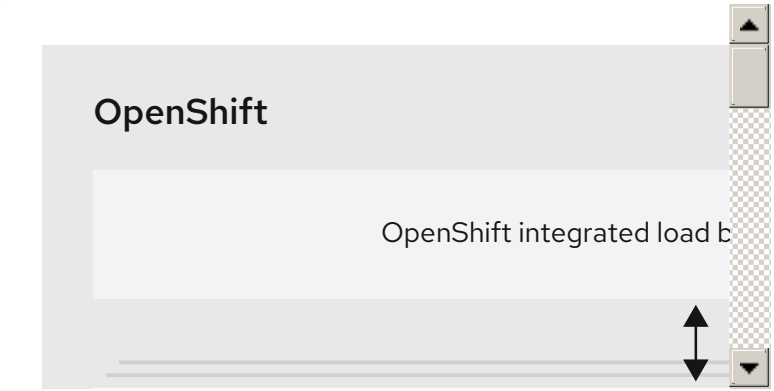


注意

OpenShift Container Platform 支持将集群部署到单个 VMware vCenter 中。不支持在多个 vCenter 上使用机器/机器集部署集群。

12.2.1. 为 vSphere 设置 VMC

您可以在 AWS 托管的 vSphere 集群上安装 OpenShift Container Platform（VMC），以便启用在混合云内部和内部管理应用程序。



在 VMware vSphere 上安装 OpenShift Container Platform 之前，您必须在 VMC 环境中配置多个选项。确定您的 VMC 环境有以下先决条件：

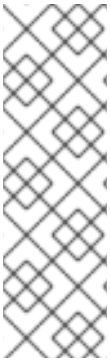
- 创建非专用的、启用了 DHCP 的、NSX-T 网络片段和子网。其他虚拟机（VM）可以托管在子网上，但至少有 8 个 IP 地址可用于 OpenShift Container Platform 部署。
- 在 DHCP 范围之外分配两个 IP 地址，并使用反向 DNS 记录配置它们。
  - **api.<cluster_name>.<base_domain>** 的 DNS 记录，指向分配的 IP 地址。
  - ***.apps.<cluster_name>.<base_domain>** 的 DNS 记录，指向分配的 IP 地址。

- 配置以下防火墙规则：
  - OpenShift Container Platform 计算网络和互联网间的 ANY:ANY 防火墙规则。节点和应用程序用于下载容器镜像。
  - 在安装主机和软件定义的数据中心（SDDC）管理网络之间的端口 443 的 ANY:ANY 防火墙规则。这可让您在部署过程中上传 Red Hat Enterprise Linux CoreOS（RHCOS）OVA。
  - OpenShift Container Platform 计算网络和 vCenter 间的 HTTPS 防火墙规则。此连接允许 OpenShift Container Platform 与 vCenter 通信，以置备和管理节点、持久性卷声明（PVC）和其他资源。
- 您必须具有以下信息才能部署 OpenShift Container Platform:
  - OpenShift Container Platform 集群名称，如 **vmc-prod-1**。
  - 基本 DNS 名称，如 **companyname.com**。
  - 如果没有使用默认设置，则必须识别 Pod 网络 CIDR 和服务网络 CIDR，默认设置为 **10.128.0.0/14** 和 **172.30.0.0/16**。这些 CIDR 用于 pod 到 pod 和 pod 到服务的通信，且无法在外部访问，但不得与机构中的现有子网重叠。
  - 以下 vCenter 信息：
    - vCenter 主机名、用户名和密码
    - 数据中心名称，如 **SDDC-Datacenter**
    - 集群名称，如 **Cluster-1**
    - 网络名称
    - 数据存储名称，如 **WorkloadDatastore**

注意

建议在集群安装完成后将 vSphere 集群移到 VMC **Compute-ResourcePool** 资源池。

- 基于 Linux 的主机作为堡垒部署到 VMC。
  - 堡垒主机可以是 Red Hat Enterprise Linux（RHEL）或其他基于 Linux 的主机，它必须具有互联网连接，并可以将 OVA 上传到 ESXi 主机。
  - 将 OpenShift CLI 工具下载并安装到堡垒主机。
    - **openshift-install** 安装程序
    - OpenShift CLI（**oc**）工具



注意

您不能将 VMware NSX Container Plugin 用于 Kubernetes（NCP），NCP 不使用 NSX 作为 OpenShift SDN。目前 VMC 提供的 NSX 版本与 OpenShift Container Platform 认证的 NCP 版本不兼容。

但是，NSX DHCP 服务会通过全堆栈自动 OpenShift Container Platform 部署来管理虚拟机 IP 管理，节点可以手动或自动置备，可通过 Machine API 与 vSphere 集成。另外，还创建 NSX 防火墙规则，以便启用 OpenShift Container Platform 集群以及堡垒主机和 VMC vSphere 主机间的访问。

12.2.1.1. VMC Sizer 工具

AWS 上的 VMware Cloud 基于 AWS 裸机基础架构构建，这与运行 AWS 原生服务的裸机基础架构相同。当在 AWS 软件定义的数据中心（SDDC）上部署 VMware 云时，您要将这些物理服务器节点以单一租户方式运行 VMware ESXi hypervisor。这意味着其他使用 VMC 的用户无法访问物理基础结构。务必要考虑托管您的虚拟基础架构所需的物理主机数量。

要确定这一点，VMware 在 AWS Sizer 上提供 VMC。使用这个工具，您可以定义要在 VMC 上托管的资源：

- 工作负载类型
- 虚拟机总数
- 规格信息，例如：
  - 存储要求
  - vCPUs
  - vRAM
  - 过量使用比例

通过这些信息，sizer 工具可以根据 VMware 最佳实践生成报告，并推荐集群配置和您需要的主机数量。

12.2.2. vSphere 先决条件

- 置备块 registry 存储。如需有关持久性存储的更多信息，请参阅了解持久性存储。
- 查看有关 OpenShift Container Platform 安装和更新流程的详细信息。
- 如果使用防火墙，则必须将其配置为允许集群需要访问的站点。

注意

如果您要配置代理，请务必也要查看此站点列表。

12.2.3. OpenShift Container Platform 的互联网访问

在 OpenShift Container Platform 4.6 中，您需要访问互联网来安装集群。

您必须具有以下互联网访问权限：

- 访问 OpenShift Cluster Manager 以下载安装程序并执行订阅管理。如果集群可以访问互联网，并且没有禁用 Telemetry，该服务会自动授权您的集群。
- 访问 Quay.io，以获取安装集群所需的软件包。
- 获取执行集群更新所需的软件包。

重要

如果您的集群无法直接访问互联网，则可以在置备的某些类基础架构上执行受限网络安装。在此过程中，您要下载所需的内容，并使用它在镜像 registry（mirror registry） 中填充安装集群并生成安装程序所需的软件包。对于某些安装类型，集群要安装到的环境不需要访问互联网。在更新集群之前，要更新 registry 镜像系统中的内容。

12.2.4. VMware vSphere 基础架构要求

您必须在满足您使用的组件要求的 VMware vSphere 版本 6 或 7 实例上安装 OpenShift Container Platform 集群。

表 12.6. VMware 组件支持的最低 vSphere 版本

组件	最低支持版本	描述
虚拟机监控程序	vSphere 6.5 及之后的版本 13	此版本是 Red Hat Enterprise Linux CoreOS(RHCOS)支持的最低版本。请查看 Red Hat Enterprise Linux 8 支持的管理程序列表。
使用 in-tree 驱动程序存储	vSphere 6.5 及之后的版本	此插件使用 OpenShift Container Platform 中包含的 vSphere 的树内存储驱动程序创建 vSphere 存储。

如果您使用 vSphere 版本 6.5 实例，请在安装 OpenShift Container Platform 前考虑升级到 6.7U3 或 7.0。

重要

您必须确保在安装 OpenShift Container Platform 前同步 ESXi 主机上的时间。请参阅 VMware 文档中的编辑主机时间配置。

12.2.5. 网络连接要求

您必须配置机器之间的网络连接，以允许 OpenShift Container Platform 集群组件进行通信。

查看有关所需网络端口的以下详细信息。

表 12.7. 用于全机器到所有机器通信的端口

协议	端口	描述
ICMP	N/A	网络可访问性测试
TCP	**1936**	指标
	9000-9999	主机级别的服务，包括端口 **9100-9101** 上的节点导出器和端口 **9099** 上的 Cluster Version Operator。
	10250-10259	Kubernetes 保留的默认端口
	10256	openshift-sdn
UDP	**4789**	虚拟可扩展 LAN(VXLAN)
	6081	Geneve
	9000-9999	主机级别的服务，包括端口 **9100-9101** 上的节点导出器。
	500	IPsec IKE 数据包
	4500	IPsec NAT-T 数据包
TCP/UDP	**30000-32767**	Kubernetes 节点端口
ESP	N/A	IPsec Encapsulating Security Payload(ESP)

表 12.8. 用于所有机器控制平面通信的端口

协议	端口	描述
TCP	**6443**	Kubernetes API

表 12.9. control plane 机器用于 control plane 机器通信的端口

协议	端口	描述
TCP	**2379-2380**	etcd 服务器和对等端口

12.2.6. vCenter 要求

在使用安装程序置备的基础架构的 vCenter 上安装 OpenShift Container Platform 集群前，您必须准备自己的环境。

所需的 vCenter 帐户权限
要在 vCenter 中安装 OpenShift Container Platform 集群，安装程序需要一个具有特权的帐户来读取和创建所需资源。使用具有全局管理特权的帐户是访问所有必要权限的最简单方式。

如果无法使用具有全局管理特权的帐户，您必须创建角色来授予 OpenShift Container Platform 集群安装所需的权限。虽然大多数权限始终是必需的，但是一些权限只有在计划安装程序需要在您的 vCenter 实例中置备一个包含 OpenShift Container Platform 集群的文件夹时（这是默认行为）才需要。您必须为指定对象创建或修改 vSphere 角色，才能授予所需的权限。

如果安装程序创建 vSphere 虚拟机文件夹，则需要额外的角色。

例 12.3. 安装所需的角色和权限

角色的 vSphere 对象	何时需要	所需的权限
vSphere vCenter	Always	**Cns.Searchable InventoryService.Tagging.AttachTag InventoryService.Tagging.CreateCategory InventoryService.Tagging.CreateTag InventoryService.Tagging.DeleteCategory InventoryService.Tagging.DeleteTag InventoryService.Tagging.EditCategory InventoryService.Tagging.EditTag Sessions.ValidateSession StorageProfile.View**
vSphere vCenter Cluster	Always	**Host.Config.Storage Resource.AssignVMToPool VApp.AssignResourcePool VApp.Import VirtualMachine.Config.AddNewDisk**
vSphere Datastore	Always	**Datastore.AllocateSpace Datastore.Browse Datastore.FileManagement**
vSphere 端口组	Always	**Network.Assign**

角色的 vSphere 对象	何时需要	所需的权限
虚拟机文件夹	Always	**Resource.AssignVMToPool** **VApp.Import** **VirtualMachine.Config.Add ExistingDisk** **VirtualMachine.Config.Add NewDisk** **VirtualMachine.Config.Add RemoveDevice** **VirtualMachine.Config.Adva ncedConfig** **VirtualMachine.Config.Anno tation** **VirtualMachine.Config.CPU Count** **VirtualMachine.Config.Disk Extend** **VirtualMachine.Config.Disk Lease** **VirtualMachine.Config.Edit Device** **VirtualMachine.Config.Mem ory** **VirtualMachine.Config.Rem oveDisk** **VirtualMachine.Config.Rena me** **VirtualMachine.Config.Rese tGuestInfo** **VirtualMachine.Config.Reso urce** **VirtualMachine.Config.Setti ngs** **VirtualMachine.Config.Upgr adeVirtualHardware** **VirtualMachine.Interact.Gue stControl** **VirtualMachine.Interact.Pow erOff** **VirtualMachine.Interact.Pow erOn** **VirtualMachine.Interact.Res et** **VirtualMachine.Inventory.Cr eate** **VirtualMachine.Inventory.Cr eateFromExisting** **VirtualMachine.Inventory.D elete** **VirtualMachine.Provisionin g.Clone**

角色的 vSphere 对象	何时需要	所需的权限
vSphere vCenter Datacenter	如果安装程序创建虚拟机文件夹	**Resource.AssignVMToPool VApp.Import VirtualMachine.Config.Add ExistingDisk VirtualMachine.Config.Add NewDisk VirtualMachine.Config.Add RemoveDevice VirtualMachine.Config.Adva ncedConfig VirtualMachine.Config.Anno tation VirtualMachine.Config.CPU Count VirtualMachine.Config.Disk Extend VirtualMachine.Config.Disk Lease VirtualMachine.Config.Edit Device VirtualMachine.Config.Mem ory VirtualMachine.Config.Rem oveDisk VirtualMachine.Config.Rena me VirtualMachine.Config.Rese tGuestInfo VirtualMachine.Config.Reso urce VirtualMachine.Config.Setti ngs VirtualMachine.Config.Upgr adeVirtualHardware VirtualMachine.Interact.Gue stControl VirtualMachine.Interact.Pow erOff VirtualMachine.Interact.Pow erOn VirtualMachine.Interact.Res et VirtualMachine.Inventory.Cr eate VirtualMachine.Inventory.Cr eateFromExisting VirtualMachine.Inventory.D elete VirtualMachine.Provisionin g.Clone Folder.Create Folder.Delete**

此外，用户需要一些 **ReadOnly** 权限，某些角色需要权限来提升对子对象的权限。这些设置会根据您是否将集群安装到现有文件夹而有所不同。

例 12.4. 所需的权限和传播设置

vSphere 对象	文件夹类型	传播到子对象	所需的权限
vSphere vCenter	Always	False	列出所需的权限
vSphere vCenter Datacenter	现有文件夹	False	**ReadOnly** 权限
	安装程序创建文件夹	True	列出所需的权限
vSphere vCenter Cluster	Always	True	列出所需的权限
vSphere vCenter Datastore	Always	False	列出所需的权限
vSphere Switch	Always	False	**ReadOnly** 权限
vSphere 端口组	Always	False	列出所需的权限
vSphere vCenter Virtual Machine Folder	现有文件夹	True	列出所需的权限

有关只使用所需权限创建帐户的更多信息，请参阅 vSphere 文档中的 vSphere 权限和用户管理任务。

将 OpenShift Container Platform 与 vMotion 搭配使用

如果要在 vSphere 环境中使用 vMotion，请在安装 OpenShift Container Platform 集群前考虑以下内容。

- OpenShift Container Platform 通常支持仅用于计算的 vMotion。使用 Storage vMotion 可能会导致问题且不被支持。
  为了帮助确保计算和 control plane 节点的正常运行时间，建议您遵循 VMware 最佳实践进行 vMotion。还建议使用 VMware 反关联性规则来改进 OpenShift Container Platform 在维护或硬件问题期间的可用性。

  有关 vMotion 和 anti-affinity 规则的更多信息，请参阅 VMware vSphere 文档 了解 vMotion 网络要求和虚拟机反关联性规则。

- 如果您在 pod 中使用 vSphere 卷，请手动或通过 Storage vMotion 在数据存储间迁移虚拟机，从而导致 OpenShift Container Platform 持久性卷(PV)对象中的无效引用。这些引用可防止受影响的 pod 启动，并可能导致数据丢失。

- 同样，OpenShift Container Platform 不支持在数据存储间有选择地迁移 VMDK、使用数据存储集群进行虚拟机置备、动态或静态置备 PV，或使用作为数据存储集群一部分的数据存储进行 PV 的动态或静态置备。

集群资源

当部署使用安装程序置备的基础架构的 OpenShift Container Platform 集群时，安装程序必须能够在 vCenter 实例中创建多个资源。

标准 OpenShift Container Platform 安装会创建以下 vCenter 资源：

- 1 个文件夹
- 1 标签（Tag）类别
- 1 个标签（Tag）
- 虚拟机:
  - 1 个模板
  - 1 个临时 bootstrap 节点
  - 3 个 control plane 节点
  - 3 个计算机器

虽然这些资源使用了 856 GB 存储，但 bootstrap 节点会在集群安装过程中被销毁。使用标准集群至少需要 800 GB 存储。

如果部署了更多计算机器，OpenShift Container Platform 集群将使用更多存储。

集群的限制
可用资源因集群而异。vCenter 中可能的集群数量主要受可用存储空间以及对所需资源数量的限制。确保考虑集群创建的 vCenter 资源的限制和部署集群所需的资源，如 IP 地址和网络。

网络要求
网络必须使用 DHCP，并确保 DHCP 服务器被配置为为集群机器提供持久的 IP 地址。

注意

在安装开始前，持久性 IP 地址不可用。分配 DHCP 范围后，在安装后使用持久 IP 地址手动替换分配。

另外，在安装 OpenShift Container Platform 集群前，必须创建以下网络资源：

注意

建议集群中的每个 OpenShift Container Platform 节点都可以访问可通过 DHCP 发现的网络时间协议（NTP）服务器。没有 NTP 服务器也可安装。但是，异步服务器时钟将导致错误，NTP 服务器会阻止。

所需的 IP 地址
安装程序置备的 vSphere 安装需要这些静态 IP 地址：

- API 地址用于访问集群 API。
- Ingress 地址用于集群入口流量。
- 当将集群从版本 4.5 升级到 4.6 时，会使用 control plane 节点地址。

安装 OpenShift Container Platform 集群时，必须向安装程序提供这些 IP 地址。

DNS 记录

您必须在正确的 DNS 服务器中为托管 OpenShift Container Platform 集群的 vCenter 实例创建两个静态 IP 地址的 DNS 记录。在每个记录中，**<cluster_name>** 是集群名称，**<base_domain>** 是您在安装集群时指定的集群基域。完整的 DNS 记录采用如下格式: **<component>.<cluster_name>.<base_domain>.**。

表 12.10. 所需的 DNS 记录

组件	记录	描述
API VIP	**api.<cluster_name>.<base_domain>.**	此 DNS A/AAAA 或 CNAME 记录必须指向 control plane 机器的负载均衡器。此记录必须能由集群外的客户端和集群内的所有节点解析。
Ingress VIP	***.apps.<cluster_name>.<base_domain>.**	通配符 DNS A/AAAA 或 CNAME 记录，指向以运行入口路由器 Pod 的机器（默认为 worker 节点）为目标的负载均衡器。此记录必须能由集群外的客户端和集群内的所有节点解析。

12.2.7. 生成 SSH 私钥并将其添加到代理中

如果要在集群上执行安装调试或灾难恢复，则必须为 **ssh-agent** 和安装程序提供 SSH 密钥。您可以使用此密钥访问公共集群中的 bootstrap 机器来排除安装问题。

注意

在生产环境中，您需要进行灾难恢复和调试。

您可以使用此密钥以 **core** 用户身份通过 SSH 连接到 master 节点。在部署集群时，此密钥会添加到 **core** 用户的 **~/.ssh/authorized_keys** 列表中。

注意

您必须使用一个本地密钥，而不要使用在特定平台上配置的密钥，如 AWS 密钥对。

流程

1. 如果还没有为计算机上免密码身份验证而配置的 SSH 密钥，请创建一个。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ ssh-keygen -t ed25519 -N '' \
  -f <path>/<file_name> 1
```

1 指定新 SSH 密钥的路径和文件名，如 **~/.ssh/id_rsa**。如果您已有密钥对，请确保您的公钥位于 **~/.ssh** 目录中。

运行此命令会在指定的位置生成不需要密码的 SSH 密钥。

注意

如果您计划在 **x86_64** 架构中安装使用 FIPS 验证的/Modules in Process 加密库的 OpenShift Container Platform 集群，不要创建使用 **ed25519** 算法的密钥。反之，创建一个使用 **rsa** 或 **ecdsa** 算法的密钥。

2. 作为后台任务启动 **ssh-agent** 进程：

```
$ eval "$(ssh-agent -s)"
```

输出示例

```
Agent pid 31874
```

注意

如果您的集群采用 FIPS 模式，则只使用 FIPS 兼容算法来生成 SSH 密钥。密钥必须是 RSA 或 ECDSA。

3. 将 SSH 私钥添加到 **ssh-agent**：

```
$ ssh-add <path>/<file_name> (1)
```

输出示例

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

(1) 指定 SSH 私钥的路径和文件名，如 **~/.ssh/id_rsa**

后续步骤

- 在安装 OpenShift Container Platform 时，为安装程序提供 SSH 公钥。

12.2.8. 获取安装程序

在安装 OpenShift Container Platform 之前，将安装文件下载到本地计算机上。

先决条件

- 运行 Linux 或 macOS 的计算机，本地磁盘空间为 500 MB

流程

1. 访问 OpenShift Cluster Manager 站点的 Infrastructure Provider 页面。如果您有红帽帐号，请使用自己的凭证登录。如果没有，请创建一个帐户。
2. 选择您的基础架构供应商。
3. 进入适用于您的安装类型的页面，下载您的操作系统的安装程序，并将文件放在要保存安装配置文件的目录中。。

重要

安装程序会在用来安装集群的计算机上创建若干文件。在完成集群安装后，您必须保留安装程序和安装程序所创建的文件。这两个文件都需要删除集群。

重要

删除安装程序创建的文件不会删除您的集群，即使集群在安装过程中失败也是如此。要删除集群，为特定云供应商完成 OpenShift Container Platform 卸载流程。

4. 提取安装程序。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ tar xvf openshift-install-linux.tar.gz
```

5. 从 Red Hat OpenShift Cluster Manager 下载安装 pull secret 。通过此 pull secret，您可以进行所含授权机构提供的服务的身份验证，这些服务包括为 OpenShift Container Platform 组件提供容器镜像的 Quay.io。

12.2.9. 在您的系统信任中添加 vCenter root CA 证书

由于安装程序需要访问 vCenter 的 API，所以必须在安装 OpenShift Container Platform 集群前将 vCenter 的可信 root CA 证书添加到系统信任中。

流程

1. 在 vCenter 主页中下载 vCenter 的 root CA 证书。在 vSphere Web Services SDK 部分点击 **Download trusted root CA certificates**。**<vCenter>/certs/download.zip** 文件下载。

2. 提取包含 vCenter root CA 证书的压缩文件。压缩文件的内容类似以下文件结构：

```
certs
├── lin
│   ├── 108f4d17.0
│   ├── 108f4d17.r1
│   ├── 7e757f6a.0
│   ├── 8e4f8471.0
│   └── 8e4f8471.r0
├── mac
│   ├── 108f4d17.0
│   ├── 108f4d17.r1
│   ├── 7e757f6a.0
│   ├── 8e4f8471.0
│   └── 8e4f8471.r0
└── win
    ├── 108f4d17.0.crt
    ├── 108f4d17.r1.crl
    ├── 7e757f6a.0.crt
    ├── 8e4f8471.0.crt
    └── 8e4f8471.r0.crl

3 directories, 15 files
```

3. 将您的操作系统的文件添加到系统信任中。例如，在 Fedora 操作系统上运行以下命令：

```
# cp certs/lin/* /etc/pki/ca-trust/source/anchors
```

4. 更新您的系统信任关系。例如，在 Fedora 操作系统上运行以下命令：

```
# update-ca-trust extract
```

12.2.10. 创建安装配置文件

您可以自定义在 VMware vSphere 上安装的 OpenShift Container Platform 集群。

先决条件

- 获取 OpenShift Container Platform 安装程序以及集群的 pull secret。

流程

1. 创建 **install-config.yaml** 文件。
   a. 更改到包含安装程序的目录，再运行以下命令：

```
$ ./openshift-install create install-config --dir <installation_directory> 1
```

   1 对于 **<installation_directory>**，请指定用于保存安装程序所创建的文件的目录名称。

重要

指定一个空目录。一些安装信息，如 bootstrap X.509 证书，有较短的过期间隔，因此不要重复使用安装目录。如果要重复使用另一个集群安装中的个别文件，可以将其复制到您的目录中。但是，一些安装数据的文件名可能会在发行版本之间有所改变。从 OpenShift Container Platform 老版本中复制安装文件时要格外小心。

   b. 在提示符处，提供您的云的配置详情：
      i. 可选：选择用来访问集群机器的 SSH 密钥。

注意

对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 **ssh-agent** 进程使用的 SSH 密钥。

      ii. 选择 **vsphere** 作为目标平台。
      iii. 指定 vCenter 实例的名称。
      iv. 指定创建集群所需的权限的 vCenter 帐户的用户名和密码。
      安装程序连接到您的 vCenter 实例。
      v. 选择要连接的 vCenter 实例中的数据中心。
      vi. 选择要使用的默认 vCenter 数据存储。

vii. 选择要在其中安装 vCenter 集群的 OpenShift Container Platform 集群。安装程序使用 vSphere 集群的 root 资源池作为默认资源池。

viii. 选择包含您配置的虚拟 IP 地址和 DNS 记录的 vCenter 实例中的网络。

ix. 输入您为 control plane API 访问配置的虚拟 IP 地址。

x. 输入您为集群入口配置的虚拟 IP 地址。

xi. 输入基域。这个基域必须与您配置的 DNS 记录中使用的域相同。

xii. 为集群输入一个描述性名称。集群名称必须与您配置的 DNS 记录中使用的相同。

xiii. 粘贴 Red Hat OpenShift Cluster Manager 中的 pull secret 。

2. 修改 **install-config.yaml** 文件。您可以在**安装配置参数**部分中找到有关可用参数的更多信息。

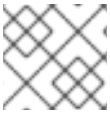
3. 备份 **install-config.yaml** 文件，以便用于安装多个集群。

重要

install-config.yaml 文件会在安装过程中消耗掉。如果要重复使用此文件，必须现在备份。

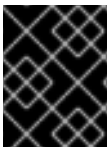
12.2.10.1. 安装配置参数

在部署 OpenShift Container Platform 集群前，您可以提供参数值，以描述托管集群的云平台的帐户并选择性地自定义集群平台。在创建 **install-config.yaml** 安装配置文件时，您可以通过命令行来提供所需的参数的值。如果要自定义集群，可以修改 **install-config.yaml** 文件来提供关于平台的更多信息。



注意

安装之后，您无法修改 **install-config.yaml** 文件中的这些参数。



重要

openshift-install 命令不验证参数的字段名称。如果指定了不正确的名称，则不会创建相关的文件或对象，且不会报告错误。确保所有指定的参数的字段名称都正确。

12.2.10.1.1. 所需的配置参数

下表描述了所需的安装配置参数：

表 12.11. 所需的参数

参数	描述	值
apiVersion	**install-config.yaml** 内容的 API 版本。当前版本是 **v1**。安装程序还可能支持旧的 API 版本。	字符串

参数	描述	值
baseDomain	云供应商的基域。此基础域用于创建到 OpenShift Container Platform 集群组件的路由。集群的完整 DNS 名称是 **baseDomain** 和 **metadata.name** 参数值的组合，其格式为 **<metadata.name>.<baseDomain>**。	完全限定域名或子域名，如 **example.com**。
metadata	Kubernetes 资源 **ObjectMeta**，其中只消耗 **name** 参数。	对象
metadata.name	集群的名称。集群的 DNS 记录是 **{{.metadata.name}}.{{.baseDomain}}** 的子域。	小写字母和连字符(**-**)的字符串，如 **dev**。
platform	执行安装的具体平台配置：**aws**、**baremetal**、**azure**、**openstack**、**ovirt**、**vsphere**。有关 **platform.<platform>** 参数的额外信息，请参考下表来了解您的具体平台。	对象

pullSecret	从 Red Hat OpenShift Cluster Manager 获取 pull secret，验证从 Quay.io 等服务中下载 OpenShift Container Platform 组件的容器镜像。	<pre>{
		  "auths":{
		  "cloud.openshift.com":{
		  "auth":"b3Blb=",
		  "email":"you@example.com"
		  },
		  "quay.io":{
		  "auth":"b3Blb=",
		  "email":"you@example.com"
		  }
		  }
		}</pre>

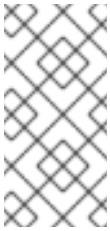
12.2.10.1.2. 网络配置参数

您可以根据现有网络基础架构的要求自定义安装配置。例如，您可以扩展集群网络的 IP 地址块，或者提供不同于默认值的不同 IP 地址块。

只支持 IPv4 地址。

表 12.12. 网络参数

参数	描述	值
networking	集群网络的配置。	对象  **注意** 您不能在安装后修改 **networking** 对象指定的参数。
networking.networkType	要安装的集群网络供应商 Container Network Interface（CNI）插件。	**OpenShiftSDN** 或 **OVNKubernetes**。默认值为 **OpenShiftSDN**。
networking.clusterNetwork	pod 的 IP 地址块。 默认值为 **10.128.0.0/14**，主机前缀为 **/23**。 如果您指定多个 IP 地址块，则块不得互相重叠。	一个对象数组。例如： networking: clusterNetwork: - cidr: 10.128.0.0/14 hostPrefix: 23
networking.clusterNetwork.cidr	使用 **networking.clusterNetwork** 时需要此项。IP 地址块。 一个 IPv4 网络。	使用 CIDR 形式的 IP 地址块。IPv4 块的前缀长度介于 **0** 到 **32** 之间。
networking.clusterNetwork.hostPrefix	分配给每个单独节点的子网前缀长度。例如，如果 **hostPrefix** 设为 **23**，则每个节点从所给的 **cidr** 中分配一个 **/23** 子网。**hostPrefix** 值 **23** 提供 510（2^(32 - 23)- 2）个 pod IP 地址。	子网前缀。 默认值为 **23**。
networking.serviceNetwork	服务的 IP 地址块。默认值为 **172.30.0.0/16**。 OpenShift SDN 和 OVN-Kubernetes 网络供应商只支持服务网络的一个 IP 地址块。	CIDR 格式具有 IP 地址块的数组。例如： networking: serviceNetwork: - 172.30.0.0/16
networking.machineNetwork	机器的 IP 地址块。 如果您指定多个 IP 地址块，则块不得互相重叠。	一个对象数组。例如： networking: machineNetwork: - cidr: 10.0.0.0/16

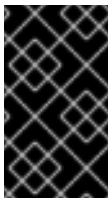
参数	描述	值
networking.machineNetwork.cidr	使用 **networking.machineNetwork** 时需要。IP 地址块。libvirt 以外的所有平台的默认值为 **10.0.0.0/16**。对于 libvirt，默认值为 **192.168.126.0/24**。	CIDR 表示法中的 IP 网络块。 例如： **10.0.0.0/16**。  **注意** 将 **networking.machineNetwork** 设置为与首选 NIC 所在的 CIDR 匹配。

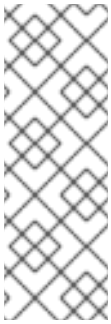
12.2.10.1.3. 可选配置参数

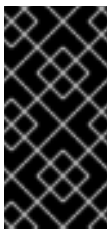
下表描述了可选安装配置参数：

表 12.13. 可选参数

参数	描述	值
additionalTrustBundle	添加到节点可信证书存储中的 PEM 编码 X.509 证书捆绑包。配置了代理时，也可以使用这个信任捆绑包。	字符串
compute	组成计算节点的机器的配置。	machine-pool 对象的数组。详情请查看以下"Machine-pool"表。
compute.architecture	决定池中机器的指令集合架构。目前不支持异构集群，因此所有池都必须指定相同的架构。有效值为 **amd64** （默认值）。	字符串
compute.hyperthreading	是否在计算机器上启用或禁用并发多线程或**超线程**。默认情况下，启用并发多线程以提高机器内核的性能。  **重要** 如果禁用并发多线程，请确保在容量规划时考虑到机器性能可能会显著降低的问题。	**Enabled** 或 **Disabled**
compute.name	使用 **compute** 时需要此值。机器池的名称。	**worker**

参数	描述	值
compute.platform	使用 **compute** 时需要此值。使用此参数指定托管 worker 机器的云供应商。此参数值必须与 **controlPlane.platform** 参数值匹配。	**aws**、**azure**、**gcp**、**openstack**、**ovirt**、**vsphere** 或 **{}**
compute.replicas	要置备的计算机器数量，也称为 worker 机器。	大于或等于 **2** 的正整数。默认值为 **3**。
controlPlane	组成 control plane 的机器的配置。	**MachinePool** 对象的数组。详情请查看以下"Machine-pool"表。
controlPlane.architecture	决定池中机器的指令集合架构。目前不支持异构集群，因此所有池都必须指定相同的架构。有效值为 **amd64** （默认值）。	字符串
controlPlane.hyperthreading	是否在 control plane 机器上启用或禁用并发多线程或**超线程**。默认情况下，启用并发多线程以提高机器内核的性能。  **重要** 如果禁用并发多线程，请确保在容量规划时考虑到机器性能可能会显著降低的问题。	**Enabled** 或 **Disabled**
controlPlane.name	使用 **controlPlane** 时需要。机器池的名称。	**master**
controlPlane.platform	使用 **controlPlane** 时需要。使用此参数指定托管 control plane 机器的云供应商。此参数值必须与 **compute.platform** 参数值匹配。	**aws**、**azure**、**gcp**、**openstack**、**ovirt**、**vsphere** 或 **{}**
controlPlane.replicas	要置备的 control plane 机器数量。	唯一支持的值是 **3**，它是默认值。

参数	描述	值
credentialsMode	Cloud Credential Operator（CCO）模式。如果没有指定任何模式，CCO 会动态地尝试决定提供的凭证的功能，在支持多个模式的平台上使用 mint 模式。  **注意** 不是所有 CCO 模式都支持所有云供应商。如需有关 CCO 模式的更多信息，请参阅*Red Hat Operator 参考指南*内容中的 *Cloud Credential Operator* 条目。	**Mint**、**Passthrough**、**Manual** 或空字符串(**""**)。
fips	启用或禁用 FIPS 模式。默认为 **false**（禁用）。如果启用了 FIPS 模式，运行 OpenShift Container Platform 的 Red Hat Enterprise Linux CoreOS (RHCOS) 机器会绕过默认的 Kubernetes 加密套件，并使用由 RHCOS 提供的加密模块。  **重要** 只有在 **x86_64** 架构中的 OpenShift Container Platform 部署支持 FIPS 验证的/Modules in Process 加密库。  **注意** 如果使用 Azure File 存储，则无法启用 FIPS 模式。	**false** 或 **true**
imageContentSources	release-image 内容的源和仓库。	对象数组。包括一个 **source** 以及可选的 **mirrors**，如下表所示。
imageContentSources.source	使用 **imageContentSources** 时需要。指定用户在镜像拉取规格中引用的仓库。	字符串
imageContentSources.mirrors	指定可能还包含同一镜像的一个或多个仓库。	字符串数组

参数	描述	值
publish	如何发布或公开集群的面向用户的端点，如 Kubernetes API、OpenShift 路由。	**Internal** 或 **External**。默认值为 **External**。 在非云平台上不支持将此字段设置为 **Internal**。  **重要** 如果将字段的值设为 **Internal**，集群将无法运行。如需更多信息，请参阅 BZ#1953035。
sshKey	用于验证集群机器访问的 SSH 密钥或密钥。  **注意** 对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 **ssh-agent** 进程使用的 SSH 密钥。	一个或多个密钥。例如： sshKey: <key1> <key2> <key3>

12.2.10.1.4. 其他 VMware vSphere 配置参数

下表描述了其他 VMware vSphere 配置参数：

表 12.14. 其他 VMware vSphere 集群参数

参数	描述	值
platform.vsphere.vCenter	vCenter 服务器的完全限定主机名或 IP 地址。	字符串
platform.vsphere.username	用于连接 vCenter 实例的用户名。此用户必须至少具有 vSphere 中 静态或动态持久性卷置备 所需的角色和权限。	字符串
platform.vsphere.password	vCenter 用户名的密码。	字符串

参数	描述	值
platform.vsphere.datacenter	要在 vCenter 实例中使用的数据中心的名称。	字符串
platform.vsphere.defaultDatastore	用于置备卷的默认数据存储名称。	字符串
platform.vsphere.folder	*可选*。安装程序创建虚拟机的现有文件夹的绝对路径。如果没有提供这个值，安装程序会创建一个文件夹，它的名称是数据中心虚拟机文件夹中的基础架构 ID。	字符串，如 **/<datacenter_name>/vm/<folder_name>/<subfolder_name>**。
platform.vsphere.network	包含您配置的虚拟 IP 地址和 DNS 记录的 vCenter 实例中的网络。	字符串
platform.vsphere.cluster	在其中安装 OpenShift Container Platform 集群的 vCenter 集群。	字符串
platform.vsphere.apiVIP	为 control plane API 访问配置的虚拟 IP（VIP）地址。	IP 地址，如 **128.0.0.1**。
platform.vsphere.ingressVIP	为集群入口配置的虚拟 IP（VIP）地址。	IP 地址，如 **128.0.0.1**。

12.2.10.1.5. 可选的 VMware vSphere 机器池配置参数

下表描述了可选的 VMware vSphere 机器池配置参数：

表 12.15. 可选的 VMware vSphere 机器池参数

参数	描述	值
platform.vsphere.clusterOSImage	安装程序从中下载 RHCOS 镜像的位置。您必须设置此参数以便在受限网络中执行安装。	HTTP 或 HTTPS URL，可选使用 SHA-256 checksum。例如：**https://mirror.openshift.com/images/rhcos-<version>-vmware.<architecture>.ova**。
platform.vsphere.osDisk.diskSizeGB	以 GB 为单位的磁盘大小。	整数
platform.vsphere.cpus	分配虚拟机的虚拟处理器内核总数。	整数

参数	描述	值
platform.vsphere.coresPerSocket	虚拟机中每个插槽的内核数。虚拟机上的虚拟套接字数量为 **platform.vsphere.cpus/platform.vsphere.coresPerSocket**。默认值为 **1**。	整数
platform.vsphere.memoryMB	以 MB 为单位的虚拟机内存大小。	整数

12.2.10.2. 安装程序置备的 VMware vSphere 集群的 install-config.yaml 文件示例

您可以自定义 install-config.yaml 文件，以指定有关 OpenShift Container Platform 集群平台的更多信息，或修改所需参数的值。

```
apiVersion: v1
baseDomain: example.com (1)
compute: (2)
- hyperthreading: Enabled (3)
  name: worker
  replicas: 3
  platform:
    vsphere: (4)
      cpus: 2
      coresPerSocket: 2
      memoryMB: 8192
      osDisk:
        diskSizeGB: 120
controlPlane: (5)
  hyperthreading: Enabled (6)
  name: master
  replicas: 3
  platform:
    vsphere: (7)
      cpus: 4
      coresPerSocket: 2
      memoryMB: 16384
      osDisk:
        diskSizeGB: 120
metadata:
  name: cluster (8)
platform:
  vsphere:
    vcenter: your.vcenter.server
    username: username
    password: password
    datacenter: datacenter
    defaultDatastore: datastore
    folder: folder
    network: VM_Network
    cluster: vsphere_cluster_name (9)
```

```
    apiVIP: api_vip
    ingressVIP: ingress_vip
fips: false
pullSecret: '{"auths": ...}'
sshKey: 'ssh-ed25519 AAAA...'
```

1 集群的基域。所有 DNS 记录都必须是这个基域的子域，并包含集群名称。

2 5 **controlPlane** 部分是一个单映射，但 **compute** 部分是一系列映射。为满足不同数据结构的要求，**compute** 部分的第一行必须以连字符 **-** 开头，**controlPlane** 部分的第一行则不可以连字符开头。只使用一个 control plane 池。

3 6 是否要启用或禁用并发多线程或**超线程**。默认情况下，启用并发多线程以提高机器内核的性能。您可以通过将参数值设为 **Disabled** 来禁用。如果您在某些集群机器上禁用并发多线程，则必须在所有集群机器上禁用。

重要

如果禁用并发多线程，请确保在容量规划时考虑到机器性能可能会显著降低的问题。如果您禁用并发多线程，则计算机必须至少使用 8 个 CPU 和 32GB RAM。

4 7 可选：为 compute 和 control plane 机器提供额外的机器池参数配置。

8 您在 DNS 记录中指定的集群名称。

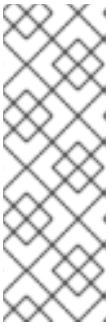
9 要在其中安装 OpenShift Container Platform 集群的 vSphere 集群。安装程序使用 vSphere 集群的 root 资源池作为默认资源池。

12.2.10.3. 在安装过程中配置集群范围代理

生产环境可能会拒绝直接访问互联网，而是提供 HTTP 或 HTTPS 代理。您可以通过在 **install-config.yaml** 文件中配置代理设置，将新的 OpenShift Container Platform 集群配置为使用代理。

先决条件

- 您有一个现有的 **install-config.yaml** 文件。
- 您检查了集群需要访问的站点，并决定是否需要绕过代理。默认情况下代理所有集群出口流量，包括对托管云供应商 API 的调用。您需要将站点添加到 **Proxy** 对象的 **spec.noProxy** 字段来绕过代理。



注意

Proxy 对象 **status.noProxy** 字段使用安装配置中的 **networking.machineNetwork[].cidr**、**networking.clusterNetwork[].cidr** 和 **networking.serviceNetwork[]** 字段的值填充。

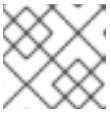
对于在 Amazon Web Services(AWS)、Google Cloud Platform(GCP)、Microsoft Azure 和 Red Hat OpenStack Platform(RHOSP)上安装, **Proxy** 对象 **status.noProxy** 字段也会使用实例元数据端点填充(**169.254.169.254**)。

流程

1. 编辑 **install-config.yaml** 文件并添加代理设置。例如：

```
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: example.com 3
additionalTrustBundle: | 4
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
...
```

1. 用于创建集群外 HTTP 连接的代理 URL。URL 必须是 **http**。
2. 用于创建集群外 HTTPS 连接的代理 URL。
3. 要排除在代理中的目标域名、IP 地址或其他网络 CIDR 的逗号分隔列表。在域前面加 **.** 来仅匹配子域。例如： **.y.com** 匹配 **x.y.com**，但不匹配 **y.com**。使用 ***** 绕过所有目的地的代理。您必须包含 vCenter 的 IP 地址以及用于其机器的 IP 范围。
4. 如果提供，安装程序会在 **openshift-config 命名空间中生成名为 user-ca- bundle** 的配置映射来保存额外的 CA 证书。如果您提供 **additionalTrustBundle** 和至少一个代理设置，则 **Proxy** 对象会被配置为引用 **trustedCA** 字段中的 **user-ca-bundle** 配置映射。然后，Cluster Network Operator 会创建一个 **trusted-ca-bundle** 配置映射，该配置映射将为 **trustedCA** 参数指定的内容与 RHCOS 信任捆绑包合并。 **additionalTrustBundle** 字段是必需的，除非代理的身份证书由来自 RHCOS 信任捆绑包的颁发机构签名。



注意

安装程序不支持代理的 **readinessEndpoints** 字段。

2. 保存该文件，并在安装 OpenShift Container Platform 时引用。

安装程序会创建一个名为 **cluster** 的集群范围代理，该代理使用提供的 **install-config.yaml** 文件中的代理设置。如果没有提供代理设置，仍然会创建一个 **cluster Proxy** 对象，但它会有一个空 **spec**。

注意

只支持名为 **cluster** 的 **Proxy** 对象，且无法创建额外的代理。

12.2.11. 部署集群

您可以在兼容云平台中安装 OpenShift Container Platform。

重要

安装程序的 **create cluster** 命令只能在初始安装过程中运行一次。

先决条件

- 配置托管集群的云平台的帐户。

- 获取 OpenShift Container Platform 安装程序以及集群的 pull secret。

流程

1. 更改为包含安装程序的目录并初始化集群部署：

```
$ ./openshift-install create cluster --dir <installation_directory> \ 1
    --log-level=info 2
```

1 对于 **<installation_directory>**，请指定自定义 **./install-config.yaml** 文件的位置。

2 要查看不同的安装详情，请指定 **warn**、**debug** 或 **error**，而不要指定 **info**。

重要

使用托管在 VMC 环境中的堡垒中的 **openshift-install** 命令。

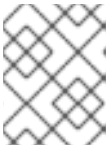
注意

如果您在主机上配置的云供应商帐户没有足够的权限来部署集群，安装过程将会停止，并且显示缺少的权限。

集群部署完成后，终端会显示访问集群的信息，包括指向其 Web 控制台的链接和 **kubeadmin** 用户的凭证。

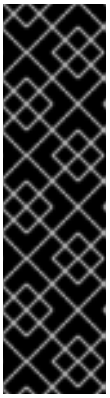
输出示例

```
...
INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run 'export
KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'
INFO Access the OpenShift web-console here: https://console-openshift-
console.apps.mycluster.example.com
INFO Login to the console with user: "kubeadmin", and password: "4vYBz-Ee6gm-ymBZj-
Wt5AL"
INFO Time elapsed: 36m22s
```



注意

当安装成功时，集群访问和凭证信息还会输出到 **<installation_directory>/.openshift_install.log**。



重要

- 安装程序生成的 Ignition 配置文件包含在 24 小时后过期的证书，然后在过期时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外情况是，您需要手动批准待处理的 **node-bootstrapper** 证书签名请求（CSR）来恢复 kubelet 证书。如需更多信息，请参阅*从过期的 control plane 证书中恢复*的文档。
- 建议您在生成 12 小时后使用 Ignition 配置文件，因为集群安装后 24 小时证书从 16 小时轮转至 22 小时。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中运行证书更新时避免安装失败。

重要

您不得删除安装程序或安装程序所创建的文件。需要这两者才能删除集群。

12.2.12. 通过下载二进制文件安装 OpenShift CLI

您需要安装 CLI（**oc**） 来使用命令行界面与 OpenShift Container Platform 进行交互。您可在 Linux 、Windows 或 macOS 上安装 **oc**。

重要

如果安装了旧版本的 **oc**，则无法使用 OpenShift Container Platform 4.6 中的所有命令。下载并安装新版本的 **oc**。

12.2.12.1. 在 Linux 上安装 OpenShift CLI

您可以按照以下流程在 Linux 上安装 OpenShift CLI（**oc**）二进制文件。

流程

1. 进入到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 在 **Version** 下拉菜单中选择相应的版本。
3. 单击 **OpenShift v4.6 Linux 客户端** 条目旁边的 **Download Now**，再保存文件。
4. 解包存档：

```
$ tar xvzf <file>
```

5. 把 **oc** 二进制代码放到 **PATH** 中的目录中。
   执行以下命令可以查看当前的 **PATH** 设置：

```
$ echo $PATH
```

安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
$ oc <command>
```

12.2.12.2. 在 Windows 上安装 OpenShift CLI

您可以按照以下流程在 Windows 上安装 OpenShift CLI（**oc**）二进制代码。

流程

1. 进入到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 在 **Version** 下拉菜单中选择相应的版本。
3. 单击 **OpenShift v4.6 Windows 客户端**条目旁边的 **Download Now**，再保存文件。
4. 使用 ZIP 程序解压存档。
5. 把 **oc** 二进制代码放到 **PATH** 中的目录中。
   要查看您的 **PATH**，请打开命令提示窗口并执行以下命令：

```
C:\> path
```

安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
C:\> oc <command>
```

12.2.12.3. 在 macOS 上安装 OpenShift CLI

您可以按照以下流程在 macOS 上安装 OpenShift CLI（**oc**）二进制代码。

流程

1. 进入到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 在 **Version** 下拉菜单中选择相应的版本。
3. 单击 **OpenShift v4.6 MacOSX 客户端**条目旁边的 **Download Now**，再保存文件。
4. 解包和解压存档。
5. 将 **oc** 二进制文件移到 PATH 的目录中。
   要查看您的 **PATH**，打开一个终端窗口并执行以下命令：

```
$ echo $PATH
```

安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
$ oc <command>
```

12.2.13. 使用 CLI 登录到集群

您可以通过导出集群 **kubeconfig** 文件，以默认系统用户身份登录集群。**kubeconfig** 文件包含关于集群的信息，供 CLI 用于将客户端连接到正确集群和 API 服务器。该文件特只适用于一个特定的集群，在 OpenShift Container Platform 安装过程中创建。

先决条件

- 已部署了 OpenShift Container Platform 集群。

- 已安装 **oc** CLI。

流程

1. 导出 **kubeadmin** 凭证：

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

2. 使用导出的配置，验证能否成功运行 **oc** 命令：

```
$ oc whoami
```

输出示例

```
system:admin
```

12.2.14. 创建 registry 存储

安装集群后，必须为 Registry Operator 创建存储。

12.2.14.1. 安装过程中删除的镜像 registry

在不提供可共享对象存储的平台上，OpenShift Image Registry Operator bootstraps 本身的状态是 **Removed**。这允许 **openshift-installer** 在这些平台类型上完成安装。

将 **ManagementState** Image Registry Operator 配置从 **Removed** 改为 **Managed**。

注意

Prometheus 控制台提供了一个 **ImageRegistryRemoved** 警报，例如：

"Image Registry has been removed.**ImageStreamTags**, **BuildConfigs** and **DeploymentConfigs** which reference **ImageStreamTags** may not work as expected.Please configure storage and update the config to **Managed** state by editing configs.imageregistry.operator.openshift.io."

12.2.14.2. 镜像 registry 存储配置

对于不提供默认存储的平台，Image Registry Operator 最初将不可用。安装后，您必须配置 registry 使用的存储，这样 Registry Operator 才可用。

示配置生产集群所需的持久性卷的说明。如果适用，显示有关将空目录配置为存储位置的说明，该位置只可用于非生产集群。

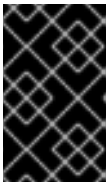
另外还提供了在升级过程中使用 **Recreate** rollout 策略来允许镜像 registry 使用块存储类型的说明。

12.2.14.2.1. 为 VMware vSphere 配置 registry 存储

作为集群管理员，在安装后需要配置 registry 来使用存储。

先决条件

- 具有 Cluster Administrator 权限
- VMware vSphere上有一个集群。
- 为集群置备的持久性存储，如 Red Hat OpenShift Container Storage。



重要

如果您只有一个副本，OpenShift Container Platform 支持对镜像 registry 存储的 **ReadWriteOnce** 访问。要部署支持高可用性的、带有两个或多个副本的镜像 registry，需要 **ReadWriteMany** 访问设置。

- 必须有“100Gi”容量。

重要

测试显示，在 RHEL 中使用 NFS 服务器作为核心服务的存储后端可能会出现问题。这包括 OpenShift Container Registry 和 Quay，Prometheus 用于监控存储，以及 Elasticsearch 用于日志存储。因此，不推荐使用 RHEL NFS 作为 PV 后端用于核心服务。

市场上的其他 NFS 实现可能没有这些问题。如需了解更多与此问题相关的信息，请联络相关的 NFS 厂商。

流程

1. 为了配置 registry 使用存储，需要修改 **configs.imageregistry/cluster** 资源中的 **spec.storage.pvc** 。

注意

使用共享存储时，请查看您的安全设置以防止被外部访问。

2. 验证您没有 registry pod:

```
$ oc get pod -n openshift-image-registry
```

注意

如果存储类型为 **emptyDIR**，则副本数不能超过 **1**。

3. 检查 registry 配置：

```
$ oc edit configs.imageregistry.operator.openshift.io
```

输出示例

```
storage:
  pvc:
    claim: 1
```

1 将 **claim** 字段留空以允许自动创建一个 **image-registry-storage** PVC。

4. 检查 **clusteroperator** 的状态：

```
$ oc get clusteroperator image-registry
```

12.2.14.2.2. 为 VMware vSphere 配置块 registry 存储

在作为集群管理员升级时，要允许镜像 registry 使用块存储类型，如 vSphere Virtual Machine Disk（VMDK），您可以使用 **Recreate** rollout 策略。

重要

支持块存储卷，但不建议将其用于生产环境中的镜像 registry。在块存储上配置 registry 的安装不具有高可用性，因为 registry 无法拥有多个副本。

流程

1. 要将镜像 registry 存储设置为块存储类型，对 registry 进行补丁，使其使用 **Recreate** rollout 策略，且仅使用 **1** 个副本运行：

```
$ oc patch config.imageregistry.operator.openshift.io/cluster --type=merge -p '{"spec":
{"rolloutStrategy":"Recreate","replicas":1}}'
```

2. 为块存储设备置备 PV，并为该卷创建 PVC。请求的块卷使用 ReadWriteOnce（RWO）访问模式。

   a. 创建包含以下内容的 **pvc.yaml** 文件以定义 VMware vSphere **PersistentVolumeClaim**：

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: image-registry-storage 1
  namespace: openshift-image-registry 2
spec:
  accessModes:
  - ReadWriteOnce 3
  resources:
    requests:
      storage: 100Gi 4
```

   1 代表 **PersistentVolumeClaim** 对象的唯一名称。

   2 **PersistentVolumeClaim** 对象的命名空间，即 **openshift-image-registry**。

   3 持久性卷声明的访问模式。使用 **ReadWriteOnce** 时，单个节点可以通过读写权限挂载这个卷。

   4 持久性卷声明的大小。

   b. 从文件创建 **PersistentVolumeClaim** 对象：

```
$ oc create -f pvc.yaml -n openshift-image-registry
```

3. 编辑 registry 配置，使其可以正确引用 PVC:

```
$ oc edit config.imageregistry.operator.openshift.io -o yaml
```

输出示例

```
storage:
  pvc:
    claim: 1
```

1 通过创建自定义 PVC，您可以将 **claim** 字段留空以用于默认自动创建 **image-registry-storage** PVC。

有关配置 registry 存储以便引用正确的 PVC 的说明，请参阅为 vSphere 配置 registry。

12.2.15. 备份 VMware vSphere 卷

OpenShift Container Platform 将新卷作为独立持久性磁盘置备，以便在集群中的任何节点上自由附加和分离卷。因此，无法备份使用快照的卷，也无法从快照中恢复卷。如需更多信息，请参阅 快照限制。

流程

要创建持久性卷的备份：

1. 停止使用持久性卷的应用程序。
2. 克隆持久性卷。
3. 重启应用程序。
4. 创建克隆的卷的备份。
5. 删除克隆的卷。

12.2.16. OpenShift Container Platform 的 Telemetry 访问

在 OpenShift Container Platform 4.6 中，默认运行的 Telemetry 服务提供有关集群健康状况和成功更新的指标，需要访问互联网。如果您的集群连接到互联网，Telemetry 会自动运行，而且集群会注册到 OpenShift Cluster Manager。

确认 OpenShift Cluster Manager 清单正确后，可以由 Telemetry 自动维护，也可以使用 OpenShift Cluster Manager 手动维护，使用订阅监控来跟踪帐户或多集群级别的 OpenShift Container Platform 订阅。

其他资源

- 有关 Telemetry 服务的更多信息，请参阅关于远程健康监控。

12.2.17. 后续步骤

- 自定义集群。

- 如果需要，您可以选择不使用远程健康报告。
- 设置 registry 并配置 registry 存储。

12.3. 使用网络自定义在 VMC 上安装集群

在 OpenShift Container Platform 版本 4.6 中，您可以使用安装程序置备的基础架构和自定义的网络配置选项在 VMware vSphere 实例上安装集群，方法是将其部署到 VMware Cloud (VMC) on AWS。

为 OpenShift Container Platform 部署配置了 VMC 环境后，您要使用堡垒管理主机中的 OpenShift Container Platform 安装程序，并位于 VMC 环境中。安装程序和 control plane 可以自动部署和管理 OpenShift Container Platform 集群所需的资源。

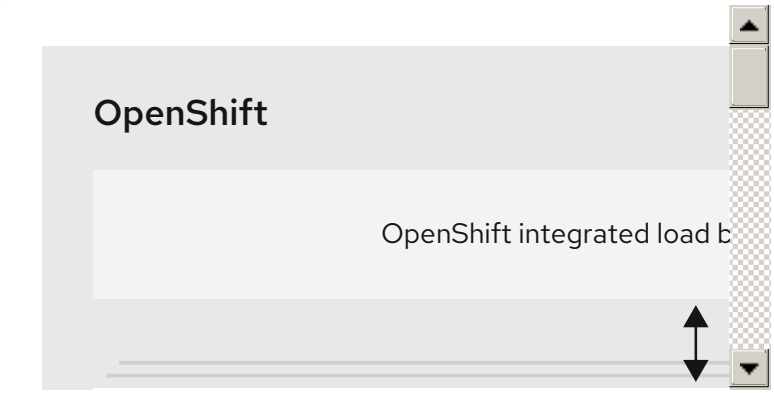
通过自定义 OpenShift Container Platform 网络配置，您的集群可以与环境中的现有 IP 地址分配共存，并与现有的 VXLAN 配置集成。要自定义安装，请在安装集群前修改 **install-config.yaml** 文件中的参数。大部分网络配置参数必须在安装过程中设置，只有 **kubeProxy** 配置参数可以在运行的集群中修改。

注意

OpenShift Container Platform 支持将集群部署到单个 VMware vCenter 中。不支持在多个 vCenter 上使用机器/机器集部署集群。

12.3.1. 为 vSphere 设置 VMC

您可以在 AWS 托管的 vSphere 集群上安装 OpenShift Container Platform（VMC），以便启用在混合云内部和内部管理应用程序。



在 VMware vSphere 上安装 OpenShift Container Platform 之前，您必须在 VMC 环境中配置多个选项。确定您的 VMC 环境有以下先决条件：

- 创建非专用的、启用了 DHCP 的、NSX-T 网络片段和子网。其他虚拟机（VM）可以托管在子网上，但至少有 8 个 IP 地址可用于 OpenShift Container Platform 部署。
- 在 DHCP 范围之外分配两个 IP 地址，并使用反向 DNS 记录配置它们。
  - **api.<cluster_name>.<base_domain>** 的 DNS 记录，指向分配的 IP 地址。
  - ***.apps.<cluster_name>.<base_domain>** 的 DNS 记录，指向分配的 IP 地址。
- 配置以下防火墙规则：
  - OpenShift Container Platform 计算网络和互联网间的 ANY:ANY 防火墙规则。节点和应用程序用于下载容器镜像。

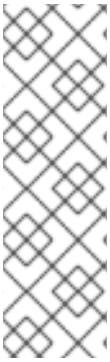
- 在安装主机和软件定义的数据中心（SDDC）管理网络之间的端口 443 的 ANY:ANY 防火墙规则。这可让您在部署过程中上传 Red Hat Enterprise Linux CoreOS（RHCOS）OVA。
- OpenShift Container Platform 计算网络和 vCenter 间的 HTTPS 防火墙规则。此连接允许 OpenShift Container Platform 与 vCenter 通信，以置备和管理节点、持久性卷声明（PVC）和其他资源。

- 您必须具有以下信息才能部署 OpenShift Container Platform:
  - OpenShift Container Platform 集群名称，如 **vmc-prod-1**。
  - 基本 DNS 名称，如 **companyname.com**。
  - 如果没有使用默认设置，则必须识别 Pod 网络 CIDR 和服务网络 CIDR，默认设置为 **10.128.0.0/14** 和 **172.30.0.0/16**。这些 CIDR 用于 pod 到 pod 和 pod 到服务的通信，且无法在外部访问，但不得与机构中的现有子网重叠。
  - 以下 vCenter 信息：
    - vCenter 主机名、用户名和密码
    - 数据中心名称，如 **SDDC-Datacenter**
    - 集群名称，如 **Cluster-1**
    - 网络名称
    - 数据存储名称，如 **WorkloadDatastore**

注意

建议在集群安装完成后将 vSphere 集群移到 VMC **Compute-ResourcePool** 资源池。

- 基于 Linux 的主机作为堡垒部署到 VMC。
  - 堡垒主机可以是 Red Hat Enterprise Linux（RHEL）或其他基于 Linux 的主机，它必须具有互联网连接，并可以将 OVA 上传到 ESXi 主机。
  - 将 OpenShift CLI 工具下载并安装到堡垒主机。
    - **openshift-install** 安装程序
    - OpenShift CLI（**oc**）工具



注意

您不能将 VMware NSX Container Plugin 用于 Kubernetes（NCP），NCP 不使用 NSX 作为 OpenShift SDN。目前 VMC 提供的 NSX 版本与 OpenShift Container Platform 认证的 NCP 版本不兼容。

但是，NSX DHCP 服务会通过全堆栈自动 OpenShift Container Platform 部署来管理虚拟机 IP 管理，节点可以手动或自动置备，可通过 Machine API 与 vSphere 集成。另外，还创建 NSX 防火墙规则，以便启用 OpenShift Container Platform 集群以及堡垒主机和 VMC vSphere 主机间的访问。

12.3.1.1. VMC Sizer 工具

AWS 上的 VMware Cloud 基于 AWS 裸机基础架构构建，这与运行 AWS 原生服务的裸机基础架构相同。当在 AWS 软件定义的数据中心（SDDC）上部署 VMware 云时，您要将这些物理服务器节点以单一租户方式运行 VMware ESXi hypervisor。这意味着其他使用 VMC 的用户无法访问物理基础结构。务必要考虑托管您的虚拟基础架构所需的物理主机数量。

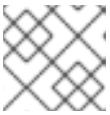
要确定这一点，VMware 在 AWS Sizer 上提供 VMC。使用这个工具，您可以定义要在 VMC 上托管的资源：

- 工作负载类型
- 虚拟机总数
- 规格信息，例如：
  - 存储要求
  - vCPUs
  - vRAM
  - 过量使用比例

通过这些信息，sizer 工具可以根据 VMware 最佳实践生成报告，并推荐集群配置和您需要的主机数量。

12.3.2. vSphere 先决条件

- 置备块 registry 存储。如需有关持久性存储的更多信息，请参阅了解持久性存储。
- 查看有关 OpenShift Container Platform 安装和更新流程的详细信息。
- 如果使用防火墙，则必须将其配置为允许集群需要访问的站点。



注意

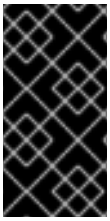
如果您要配置代理，请务必也要查看此站点列表。

12.3.3. OpenShift Container Platform 的互联网访问

在 OpenShift Container Platform 4.6 中，您需要访问互联网来安装集群。

您必须具有以下互联网访问权限：

- 访问 OpenShift Cluster Manager 以下载安装程序并执行订阅管理。如果集群可以访问互联网，并且没有禁用 Telemetry，该服务会自动授权您的集群。
- 访问 Quay.io，以获取安装集群所需的软件包。
- 获取执行集群更新所需的软件包。



重要

如果您的集群无法直接访问互联网，则可以在置备的某些类基础架构上执行受限网络安装。在此过程中，您要下载所需的内容，并使用它在镜像 registry（mirror registry） 中填充安装集群并生成安装程序所需的软件包。对于某些安装类型，集群要安装到的环境不需要访问互联网。在更新集群之前，要更新 registry 镜像系统中的内容。

12.3.4. VMware vSphere 基础架构要求

您必须在满足您使用的组件要求的 VMware vSphere 版本 6 或 7 实例上安装 OpenShift Container Platform 集群。

表 12.16. VMware 组件支持的最低 vSphere 版本

组件	最低支持版本	描述
虚拟机监控程序	vSphere 6.5 及之后的版本 13	此版本是 Red Hat Enterprise Linux CoreOS(RHCOS)支持的最低版本。请查看 Red Hat Enterprise Linux 8 支持的管理程序列表。
使用 in-tree 驱动程序存储	vSphere 6.5 及之后的版本	此插件使用 OpenShift Container Platform 中包含的 vSphere 的树内存储驱动程序创建 vSphere 存储。

如果您使用 vSphere 版本 6.5 实例，请在安装 OpenShift Container Platform 前考虑升级到 6.7U3 或 7.0。

重要

您必须确保在安装 OpenShift Container Platform 前同步 ESXi 主机上的时间。请参阅 VMware 文档中的编辑主机时间配置。

12.3.5. 网络连接要求

您必须配置机器之间的网络连接，以允许 OpenShift Container Platform 集群组件进行通信。

查看有关所需网络端口的以下详细信息。

表 12.17. 用于全机器到所有机器通信的端口

协议	端口	描述
ICMP	N/A	网络可访问性测试
TCP	**1936**	指标
	9000-9999	主机级别的服务，包括端口 **9100-9101** 上的节点导出器和端口 **9099** 上的 Cluster Version Operator。

协议	端口	描述
	10250-10259	Kubernetes 保留的默认端口
	10256	openshift-sdn
UDP	**4789**	虚拟可扩展 LAN(VXLAN)
	6081	Geneve
	9000-9999	主机级别的服务，包括端口 **9100-9101** 上的节点导出器。
	500	IPsec IKE 数据包
	4500	IPsec NAT-T 数据包
TCP/UDP	**30000-32767**	Kubernetes 节点端口
ESP	N/A	IPsec Encapsulating Security Payload(ESP)

表 12.18. 用于所有机器控制平面通信的端口

协议	端口	描述
TCP	**6443**	Kubernetes API

表 12.19. control plane 机器用于 control plane 机器通信的端口

协议	端口	描述
TCP	**2379-2380**	etcd 服务器和对等端口

12.3.6. vCenter 要求

在使用安装程序置备的基础架构的 vCenter 上安装 OpenShift Container Platform 集群前，您必须准备自己的环境。

所需的 vCenter 帐户权限
要在 vCenter 中安装 OpenShift Container Platform 集群，安装程序需要一个具有特权的帐户来读取和创建所需资源。使用具有全局管理特权的帐户是访问所有必要权限的最简单方式。

如果无法使用具有全局管理特权的帐户，您必须创建角色来授予 OpenShift Container Platform 集群安装所需的权限。虽然大多数权限始终是必需的，但是一些权限只有在计划安装程序需要在您的 vCenter 实例中置备一个包含 OpenShift Container Platform 集群的文件夹时（这是默认行为）才需要。您必须为指定对象创建或修改 vSphere 角色，才能授予所需的权限。

如果安装程序创建 vSphere 虚拟机文件夹，则需要额外的角色。

例 12.5. 安装所需的角色和权限

角色的 vSphere 对象	何时需要	所需的权限
vSphere vCenter	Always	**Cns.Searchable** **InventoryService.Tagging.AttachTag** **InventoryService.Tagging.CreateCategory** **InventoryService.Tagging.CreateTag** **InventoryService.Tagging.DeleteCategory** **InventoryService.Tagging.DeleteTag** **InventoryService.Tagging.EditCategory** **InventoryService.Tagging.EditTag** **Sessions.ValidateSession** **StorageProfile.View**
vSphere vCenter Cluster	Always	**Host.Config.Storage** **Resource.AssignVMToPool** **VApp.AssignResourcePool** **VApp.Import** **VirtualMachine.Config.AddNewDisk**
vSphere Datastore	Always	**Datastore.AllocateSpace** **Datastore.Browse** **Datastore.FileManagement**
vSphere 端口组	Always	**Network.Assign**

角色的 vSphere 对象	何时需要	所需的权限
虚拟机文件夹	Always	**Resource.AssignVMToPool VApp.Import VirtualMachine.Config.Add ExistingDisk VirtualMachine.Config.Add NewDisk VirtualMachine.Config.Add RemoveDevice VirtualMachine.Config.Adva ncedConfig VirtualMachine.Config.Anno tation VirtualMachine.Config.CPU Count VirtualMachine.Config.Disk Extend VirtualMachine.Config.Disk Lease VirtualMachine.Config.Edit Device VirtualMachine.Config.Mem ory VirtualMachine.Config.Rem oveDisk VirtualMachine.Config.Rena me VirtualMachine.Config.Rese tGuestInfo VirtualMachine.Config.Reso urce VirtualMachine.Config.Setti ngs VirtualMachine.Config.Upgr adeVirtualHardware VirtualMachine.Interact.Gue stControl VirtualMachine.Interact.Pow erOff VirtualMachine.Interact.Pow erOn VirtualMachine.Interact.Res et VirtualMachine.Inventory.Cr eate VirtualMachine.Inventory.Cr eateFromExisting VirtualMachine.Inventory.D elete VirtualMachine.Provisionin g.Clone**

角色的 vSphere 对象	何时需要	所需的权限
vSphere vCenter Datacenter	如果安装程序创建虚拟机文件夹	**Resource.AssignVMToPool VApp.Import VirtualMachine.Config.Add ExistingDisk VirtualMachine.Config.Add NewDisk VirtualMachine.Config.Add RemoveDevice VirtualMachine.Config.Adva ncedConfig VirtualMachine.Config.Anno tation VirtualMachine.Config.CPU Count VirtualMachine.Config.Disk Extend VirtualMachine.Config.Disk Lease VirtualMachine.Config.Edit Device VirtualMachine.Config.Mem ory VirtualMachine.Config.Rem oveDisk VirtualMachine.Config.Rena me VirtualMachine.Config.Rese tGuestInfo VirtualMachine.Config.Reso urce VirtualMachine.Config.Setti ngs VirtualMachine.Config.Upgr adeVirtualHardware VirtualMachine.Interact.Gue stControl VirtualMachine.Interact.Pow erOff VirtualMachine.Interact.Pow erOn VirtualMachine.Interact.Res et VirtualMachine.Inventory.Cr eate VirtualMachine.Inventory.Cr eateFromExisting VirtualMachine.Inventory.D elete VirtualMachine.Provisionin g.Clone Folder.Create Folder.Delete**

此外，用户需要一些 **ReadOnly** 权限，某些角色需要权限来提升对子对象的权限。这些设置会根据您是否将集群安装到现有文件夹而有所不同。

例 12.6. 所需的权限和传播设置

vSphere 对象	文件夹类型	传播到子对象	所需的权限
vSphere vCenter	Always	False	列出所需的权限
vSphere vCenter Datacenter	现有文件夹	False	**ReadOnly** 权限
	安装程序创建文件夹	True	列出所需的权限
vSphere vCenter Cluster	Always	True	列出所需的权限
vSphere vCenter Datastore	Always	False	列出所需的权限
vSphere Switch	Always	False	**ReadOnly** 权限
vSphere 端口组	Always	False	列出所需的权限
vSphere vCenter Virtual Machine Folder	现有文件夹	True	列出所需的权限

有关只使用所需权限创建帐户的更多信息，请参阅 vSphere 文档中的 vSphere 权限和用户管理任务。

将 OpenShift Container Platform 与 vMotion 搭配使用

如果要在 vSphere 环境中使用 vMotion，请在安装 OpenShift Container Platform 集群前考虑以下内容。

- OpenShift Container Platform 通常支持仅用于计算的 vMotion。使用 Storage vMotion 可能会导致问题且不被支持。
  为了帮助确保计算和 control plane 节点的正常运行时间，建议您遵循 VMware 最佳实践进行 vMotion。还建议使用 VMware 反关联性规则来改进 OpenShift Container Platform 在维护或硬件问题期间的可用性。

  有关 vMotion 和 anti-affinity 规则的更多信息，请参阅 VMware vSphere 文档 了解 vMotion 网络要求和虚拟机反关联性规则。

- 如果您在 pod 中使用 vSphere 卷，请手动或通过 Storage vMotion 在数据存储间迁移虚拟机，从而导致 OpenShift Container Platform 持久性卷(PV)对象中的无效引用。这些引用可防止受影响的 pod 启动，并可能导致数据丢失。

- 同样，OpenShift Container Platform 不支持在数据存储间有选择地迁移 VMDK、使用数据存储集群进行虚拟机置备、动态或静态置备 PV，或使用作为数据存储集群一部分的数据存储进行 PV 的动态或静态置备。

集群资源

当部署使用安装程序置备的基础架构的 OpenShift Container Platform 集群时，安装程序必须能够在 vCenter 实例中创建多个资源。

标准 OpenShift Container Platform 安装会创建以下 vCenter 资源：

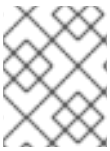
- 1 个文件夹
- 1 标签（Tag）类别
- 1 个标签（Tag）
- 虚拟机:
  - 1 个模板
  - 1 个临时 bootstrap 节点
  - 3 个 control plane 节点
  - 3 个计算机器

虽然这些资源使用了 856 GB 存储，但 bootstrap 节点会在集群安装过程中被销毁。使用标准集群至少需要 800 GB 存储。

如果部署了更多计算机器，OpenShift Container Platform 集群将使用更多存储。

集群的限制
可用资源因集群而异。vCenter 中可能的集群数量主要受可用存储空间以及对所需资源数量的限制。确保考虑集群创建的 vCenter 资源的限制和部署集群所需的资源，如 IP 地址和网络。

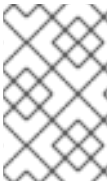
网络要求
网络必须使用 DHCP，并确保 DHCP 服务器被配置为为集群机器提供持久的 IP 地址。



注意

在安装开始前，持久性 IP 地址不可用。分配 DHCP 范围后，在安装后使用持久 IP 地址手动替换分配。

另外，在安装 OpenShift Container Platform 集群前，必须创建以下网络资源：



注意

建议集群中的每个 OpenShift Container Platform 节点都可以访问可通过 DHCP 发现的网络时间协议（NTP）服务器。没有 NTP 服务器也可安装。但是，异步服务器时钟将导致错误，NTP 服务器会阻止。

所需的 IP 地址
安装程序置备的 vSphere 安装需要这些静态 IP 地址：

- API 地址用于访问集群 API。
- Ingress 地址用于集群入口流量。
- 当将集群从版本 4.5 升级到 4.6 时，会使用 control plane 节点地址。

安装 OpenShift Container Platform 集群时，必须向安装程序提供这些 IP 地址。

DNS 记录

您必须在正确的 DNS 服务器中为托管 OpenShift Container Platform 集群的 vCenter 实例创建两个静态 IP 地址的 DNS 记录。在每个记录中，**<cluster_name>** 是集群名称，**<base_domain>** 是您在安装集群时指定的集群基域。完整的 DNS 记录采用如下格式: **<component>.<cluster_name>.<base_domain>.**。

表 12.20. 所需的 DNS 记录

组件	记录	描述
API VIP	**api.<cluster_name>.<base_domain>.**	此 DNS A/AAAA 或 CNAME 记录必须指向 control plane 机器的负载均衡器。此记录必须能由集群外的客户端和集群内的所有节点解析。
Ingress VIP	***.apps.<cluster_name>.<base_domain>.**	通配符 DNS A/AAAA 或 CNAME 记录，指向以运行入口路由器 Pod 的机器（默认为 worker 节点）为目标的负载均衡器。此记录必须能由集群外的客户端和集群内的所有节点解析。

12.3.7. 生成 SSH 私钥并将其添加到代理中

如果要在集群上执行安装调试或灾难恢复，则必须为 **ssh-agent** 和安装程序提供 SSH 密钥。您可以使用此密钥访问公共集群中的 bootstrap 机器来排除安装问题。

注意

在生产环境中，您需要进行灾难恢复和调试。

您可以使用此密钥以 **core** 用户身份通过 SSH 连接到 master 节点。在部署集群时，此密钥会添加到 **core** 用户的 **~/.ssh/authorized_keys** 列表中。

注意

您必须使用一个本地密钥，而不要使用在特定平台上配置的密钥，如 AWS 密钥对。

流程

1. 如果还没有为计算机上免密码身份验证而配置的 SSH 密钥，请创建一个。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ ssh-keygen -t ed25519 -N '' \
  -f <path>/<file_name> 1
```

1 指定新 SSH 密钥的路径和文件名，如 **~/.ssh/id_rsa**。如果您已有密钥对，请确保您的公钥位于 **~/.ssh** 目录中。

运行此命令会在指定的位置生成不需要密码的 SSH 密钥。

注意

如果您计划在 **x86_64** 架构中安装使用 FIPS 验证的/Modules in Process 加密库的 OpenShift Container Platform 集群，不要创建使用 **ed25519** 算法的密钥。反之，创建一个使用 **rsa** 或 **ecdsa** 算法的密钥。

2. 作为后台任务启动 **ssh-agent** 进程：

```
$ eval "$(ssh-agent -s)"
```

输出示例

```
Agent pid 31874
```

注意

如果您的集群采用 FIPS 模式，则只使用 FIPS 兼容算法来生成 SSH 密钥。密钥必须是 RSA 或 ECDSA。

3. 将 SSH 私钥添加到 **ssh-agent**：

```
$ ssh-add <path>/<file_name> (1)
```

输出示例

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

(1) 指定 SSH 私钥的路径和文件名，如 **~/.ssh/id_rsa**

后续步骤

- 在安装 OpenShift Container Platform 时，为安装程序提供 SSH 公钥。

12.3.8. 获取安装程序

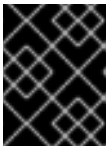
在安装 OpenShift Container Platform 之前，将安装文件下载到本地计算机上。

先决条件

- 运行 Linux 或 macOS 的计算机，本地磁盘空间为 500 MB

流程

1. 访问 OpenShift Cluster Manager 站点的 Infrastructure Provider 页面。如果您有红帽帐号，请使用自己的凭证登录。如果没有，请创建一个帐户。
2. 选择您的基础架构供应商。
3. 进入适用于您的安装类型的页面，下载您的操作系统的安装程序，并将文件放在要保存安装配置文件的目录中。。



重要

安装程序会在用来安装集群的计算机上创建若干文件。在完成集群安装后，您必须保留安装程序和安装程序所创建的文件。这两个文件都需要删除集群。

重要

删除安装程序创建的文件不会删除您的集群，即使集群在安装过程中失败也是如此。要删除集群，为特定云供应商完成 OpenShift Container Platform 卸载流程。

4. 提取安装程序。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ tar xvf openshift-install-linux.tar.gz
```

5. 从 Red Hat OpenShift Cluster Manager 下载安装 pull secret 。通过此 pull secret，您可以进行所含授权机构提供的服务的身份验证，这些服务包括为 OpenShift Container Platform 组件提供容器镜像的 Quay.io。

12.3.9. 在您的系统信任中添加 vCenter root CA 证书

由于安装程序需要访问 vCenter 的 API，所以必须在安装 OpenShift Container Platform 集群前将 vCenter 的可信 root CA 证书添加到系统信任中。

流程

1. 在 vCenter 主页中下载 vCenter 的 root CA 证书。在 vSphere Web Services SDK 部分点击 **Download trusted root CA certificates**。**<vCenter>/certs/download.zip** 文件下载。

2. 提取包含 vCenter root CA 证书的压缩文件。压缩文件的内容类似以下文件结构：

```
certs
├── lin
│   ├── 108f4d17.0
│   ├── 108f4d17.r1
│   ├── 7e757f6a.0
│   ├── 8e4f8471.0
│   └── 8e4f8471.r0
├── mac
│   ├── 108f4d17.0
│   ├── 108f4d17.r1
│   ├── 7e757f6a.0
│   ├── 8e4f8471.0
│   └── 8e4f8471.r0
└── win
    ├── 108f4d17.0.crt
    ├── 108f4d17.r1.crl
    ├── 7e757f6a.0.crt
    ├── 8e4f8471.0.crt
    └── 8e4f8471.r0.crl

3 directories, 15 files
```

3. 将您的操作系统的文件添加到系统信任中。例如，在 Fedora 操作系统上运行以下命令：

```
# cp certs/lin/* /etc/pki/ca-trust/source/anchors
```

4. 更新您的系统信任关系。例如，在 Fedora 操作系统上运行以下命令：

```
# update-ca-trust extract
```

12.3.10. 创建安装配置文件

您可以自定义在 VMware vSphere 上安装的 OpenShift Container Platform 集群。

先决条件

- 获取 OpenShift Container Platform 安装程序以及集群的 pull secret。

流程

1. 创建 **install-config.yaml** 文件。
   a. 更改到包含安装程序的目录，再运行以下命令：

```
$ ./openshift-install create install-config --dir <installation_directory> 1
```

   1 对于 **<installation_directory>**，请指定用于保存安装程序所创建的文件的目录名称。

重要

指定一个空目录。一些安装信息，如 bootstrap X.509 证书，有较短的过期间隔，因此不要重复使用安装目录。如果要重复使用另一个集群安装中的个别文件，可以将其复制到您的目录中。但是，一些安装数据的文件名可能会在发行版本之间有所改变。从 OpenShift Container Platform 老版本中复制安装文件时要格外小心。

   b. 在提示符处，提供您的云的配置详情：
      i. 可选：选择用来访问集群机器的 SSH 密钥。

注意

对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 **ssh-agent** 进程使用的 SSH 密钥。

      ii. 选择 **vsphere** 作为目标平台。
      iii. 指定 vCenter 实例的名称。
      iv. 指定创建集群所需的权限的 vCenter 帐户的用户名和密码。
      安装程序连接到您的 vCenter 实例。
      v. 选择要连接的 vCenter 实例中的数据中心。
      vi. 选择要使用的默认 vCenter 数据存储。

vii. 选择要在其中安装 vCenter 集群的 OpenShift Container Platform 集群。安装程序使用 vSphere 集群的 root 资源池作为默认资源池。

viii. 选择包含您配置的虚拟 IP 地址和 DNS 记录的 vCenter 实例中的网络。

ix. 输入您为 control plane API 访问配置的虚拟 IP 地址。

x. 输入您为集群入口配置的虚拟 IP 地址。

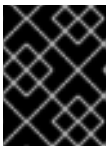
xi. 输入基域。这个基域必须与您配置的 DNS 记录中使用的域相同。

xii. 为集群输入一个描述性名称。集群名称必须与您配置的 DNS 记录中使用的相同。

xiii. 粘贴 Red Hat OpenShift Cluster Manager 中的 pull secret 。

2. 修改 **install-config.yaml** 文件。您可以在**安装配置参数**部分中找到有关可用参数的更多信息。

3. 备份 **install-config.yaml** 文件，以便用于安装多个集群。



重要

install-config.yaml 文件会在安装过程中消耗掉。如果要重复使用此文件，必须现在备份。

12.3.10.1. 安装配置参数

在部署 OpenShift Container Platform 集群前，您可以提供参数值，以描述托管集群的云平台的帐户并选择性地自定义集群平台。在创建 **install-config.yaml** 安装配置文件时，您可以通过命令行来提供所需的参数的值。如果要自定义集群，可以修改 **install-config.yaml** 文件来提供关于平台的更多信息。

注意

安装之后，您无法修改 **install-config.yaml** 文件中的这些参数。

重要

openshift-install 命令不验证参数的字段名称。如果指定了不正确的名称，则不会创建相关的文件或对象，且不会报告错误。确保所有指定的参数的字段名称都正确。

12.3.10.1.1. 所需的配置参数

下表描述了所需的安装配置参数：

表 12.21. 所需的参数

参数	描述	值
apiVersion	**install-config.yaml** 内容的 API 版本。当前版本是 **v1**。安装程序还可能支持旧的 API 版本。	字符串

参数	描述	值
baseDomain	云供应商的基域。此基础域用于创建到 OpenShift Container Platform 集群组件的路由。集群的完整 DNS 名称是 **baseDomain** 和 **metadata.name** 参数值的组合，其格式为 **<metadata.name>.<baseDomain>**。	完全限定域名或子域名，如 **example.com**。
metadata	Kubernetes 资源 **ObjectMeta**，其中只消耗 **name** 参数。	对象
metadata.name	集群的名称。集群的 DNS 记录是 **{{.metadata.name}}.{{.baseDomain}}** 的子域。	小写字母和连字符(**-**)的字符串，如 **dev**。
platform	执行安装的具体平台配置：**aws**、**baremetal**、**azure**、**openstack**、**ovirt**、**vsphere**。有关 **platform.<platform>** 参数的额外信息，请参考下表来了解您的具体平台。	对象

pullSecret	从 Red Hat OpenShift Cluster Manager 获取 pull secret，验证从 Quay.io 等服务中下载 OpenShift Container Platform 组件的容器镜像。	<pre>{
		  "auths":{
		  "cloud.openshift.com":{
		  "auth":"b3Blb=",
		  "email":"you@example.com"
		  },
		  "quay.io":{
		  "auth":"b3Blb=",
		  "email":"you@example.com"
		  }
		  }
		}</pre>

12.3.10.1.2. 网络配置参数

您可以根据现有网络基础架构的要求自定义安装配置。例如，您可以扩展集群网络的 IP 地址块，或者提供不同于默认值的不同 IP 地址块。

只支持 IPv4 地址。

表 12.22. 网络参数

参数	描述	值
networking	集群网络的配置。	对象  **注意** 您不能在安装后修改 **networking** 对象指定的参数。
networking.networkType	要安装的集群网络供应商 Container Network Interface（CNI）插件。	**OpenShiftSDN** 或 **OVNKubernetes**。默认值为 **OpenShiftSDN**。
networking.clusterNetwork	pod 的 IP 地址块。 默认值为 **10.128.0.0/14**，主机前缀为 **/23**。 如果您指定多个 IP 地址块，则块不得互相重叠。	一个对象数组。例如： networking: clusterNetwork: - cidr: 10.128.0.0/14 hostPrefix: 23
networking.clusterNetwork.cidr	使用 **networking.clusterNetwork** 时需要此项。IP 地址块。 一个 IPv4 网络。	使用 CIDR 形式的 IP 地址块。IPv4 块的前缀长度介于 **0** 到 **32** 之间。
networking.clusterNetwork.hostPrefix	分配给每个单独节点的子网前缀长度。例如，如果 **hostPrefix** 设为 **23**，则每个节点从所给的 **cidr** 中分配一个 **/23** 子网。**hostPrefix** 值 **23** 提供 510（2^(32 - 23)- 2）个 pod IP 地址。	子网前缀。 默认值为 **23**。
networking.serviceNetwork	服务的 IP 地址块。默认值为 **172.30.0.0/16**。 OpenShift SDN 和 OVN-Kubernetes 网络供应商只支持服务网络的一个 IP 地址块。	CIDR 格式具有 IP 地址块的数组。例如： networking: serviceNetwork: - 172.30.0.0/16
networking.machineNetwork	机器的 IP 地址块。 如果您指定多个 IP 地址块，则块不得互相重叠。	一个对象数组。例如： networking: machineNetwork: - cidr: 10.0.0.0/16

参数	描述	值
networking.machineNetwork.cidr	使用 **networking.machineNetwork** 时需要。IP 地址块。libvirt 以外的所有平台的默认值为 **10.0.0.0/16**。对于 libvirt，默认值为 **192.168.126.0/24**。	CIDR 表示法中的 IP 网络块。 例如： **10.0.0.0/16**。  **注意** 将 **networking.machineNetwork** 设置为与首选 NIC 所在的 CIDR 匹配。

12.3.10.1.3. 可选配置参数

下表描述了可选安装配置参数：

表 12.23. 可选参数

参数	描述	值
additionalTrustBundle	添加到节点可信证书存储中的 PEM 编码 X.509 证书捆绑包。配置了代理时，也可以使用这个信任捆绑包。	字符串
compute	组成计算节点的机器的配置。	machine-pool 对象的数组。详情请查看以下"Machine-pool"表。
compute.architecture	决定池中机器的指令集合架构。目前不支持异构集群，因此所有池都必须指定相同的架构。有效值为 **amd64** （默认值）。	字符串
compute.hyperthreading	是否在计算机器上启用或禁用并发多线程或**超线程**。默认情况下，启用并发多线程以提高机器内核的性能。  **重要** 如果禁用并发多线程，请确保在容量规划时考虑到机器性能可能会显著降低的问题。	**Enabled** 或 **Disabled**
compute.name	使用 **compute** 时需要此值。机器池的名称。	**worker**

参数	描述	值
compute.platform	使用 **compute** 时需要此值。使用此参数指定托管 worker 机器的云供应商。此参数值必须与 **controlPlane.platform** 参数值匹配。	**aws**、**azure**、**gcp**、**openstack**、**ovirt**、**vsphere** 或 **{}**
compute.replicas	要置备的计算机器数量，也称为 worker 机器。	大于或等于 **2** 的正整数。默认值为 **3**。
controlPlane	组成 control plane 的机器的配置。	**MachinePool** 对象的数组。详情请查看以下"Machine-pool"表。
controlPlane.architecture	决定池中机器的指令集合架构。目前不支持异构集群，因此所有池都必须指定相同的架构。有效值为 **amd64** （默认值）。	字符串
controlPlane.hyperthreading	是否在 control plane 机器上启用或禁用并发多线程或**超线程**。默认情况下，启用并发多线程以提高机器内核的性能。  **重要** 如果禁用并发多线程，请确保在容量规划时考虑到机器性能可能会显著降低的问题。	**Enabled** 或 **Disabled**
controlPlane.name	使用 **controlPlane** 时需要。机器池的名称。	**master**
controlPlane.platform	使用 **controlPlane** 时需要。使用此参数指定托管 control plane 机器的云供应商。此参数值必须与 **compute.platform** 参数值匹配。	**aws**、**azure**、**gcp**、**openstack**、**ovirt**、**vsphere** 或 **{}**
controlPlane.replicas	要置备的 control plane 机器数量。	唯一支持的值是 **3**，它是默认值。

参数	描述	值
credentialsMode	Cloud Credential Operator（CCO）模式。如果没有指定任何模式，CCO 会动态地尝试决定提供的凭证的功能，在支持多个模式的平台上使用 mint 模式。  **注意** 不是所有 CCO 模式都支持所有云供应商。如需有关 CCO 模式的更多信息，请参阅*Red Hat Operator 参考指南*内容中的 *Cloud Credential Operator* 条目。	**Mint**、**Passthrough**、**Manual** 或空字符串(**""**)。
fips	启用或禁用 FIPS 模式。默认为 **false**（禁用）。如果启用了 FIPS 模式，运行 OpenShift Container Platform 的 Red Hat Enterprise Linux CoreOS (RHCOS) 机器会绕过默认的 Kubernetes 加密套件，并使用由 RHCOS 提供的加密模块。  **重要** 只有在 **x86_64** 架构中的 OpenShift Container Platform 部署支持 FIPS 验证的/Modules in Process 加密库。  **注意** 如果使用 Azure File 存储，则无法启用 FIPS 模式。	**false** 或 **true**
imageContentSources	release-image 内容的源和仓库。	对象数组。包括一个 **source** 以及可选的 **mirrors**，如下表所示。
imageContentSources.source	使用 **imageContentSources** 时需要。指定用户在镜像拉取规格中引用的仓库。	字符串
imageContentSources.mirrors	指定可能还包含同一镜像的一个或多个仓库。	字符串数组

参数	描述	值
publish	如何发布或公开集群的面向用户的端点，如 Kubernetes API、OpenShift 路由。	**Internal** 或 **External**。默认值为 **External**。 在非云平台上不支持将此字段设置为 **Internal**。  **重要** 如果将字段的值设为 **Internal**，集群将无法运行。如需更多信息，请参阅 BZ#1953035。
sshKey	用于验证集群机器访问的 SSH 密钥或密钥。  **注意** 对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 **ssh-agent** 进程使用的 SSH 密钥。	一个或多个密钥。例如： sshKey: <key1> <key2> <key3>

12.3.10.1.4. 其他 VMware vSphere 配置参数

下表描述了其他 VMware vSphere 配置参数：

表 12.24. 其他 VMware vSphere 集群参数

参数	描述	值
platform.vsphere.vCenter	vCenter 服务器的完全限定主机名或 IP 地址。	字符串
platform.vsphere.username	用于连接 vCenter 实例的用户名。此用户必须至少具有 vSphere 中 静态或动态持久性卷置备 所需的角色和权限。	字符串
platform.vsphere.password	vCenter 用户名的密码。	字符串

参数	描述	值
platform.vsphere.datacenter	要在 vCenter 实例中使用的数据中心的名称。	字符串
platform.vsphere.defaultDatastore	用于置备卷的默认数据存储名称。	字符串
platform.vsphere.folder	*可选*。安装程序创建虚拟机的现有文件夹的绝对路径。如果没有提供这个值，安装程序会创建一个文件夹，它的名称是数据中心虚拟机文件夹中的基础架构 ID。	字符串，如 **/<datacenter_name>/vm/<folder_name>/<subfolder_name>**。
platform.vsphere.network	包含您配置的虚拟 IP 地址和 DNS 记录的 vCenter 实例中的网络。	字符串
platform.vsphere.cluster	在其中安装 OpenShift Container Platform 集群的 vCenter 集群。	字符串
platform.vsphere.apiVIP	为 control plane API 访问配置的虚拟 IP（VIP）地址。	IP 地址，如 **128.0.0.1**。
platform.vsphere.ingressVIP	为集群入口配置的虚拟 IP（VIP）地址。	IP 地址，如 **128.0.0.1**。

12.3.10.1.5. 可选的 VMware vSphere 机器池配置参数

下表描述了可选的 VMware vSphere 机器池配置参数：

表 12.25. 可选的 VMware vSphere 机器池参数

参数	描述	值
platform.vsphere.clusterOSImage	安装程序从中下载 RHCOS 镜像的位置。您必须设置此参数以便在受限网络中执行安装。	HTTP 或 HTTPS URL，可选使用 SHA-256 checksum。例如：**https://mirror.openshift.com/images/rhcos-<version>-vmware.<architecture>.ova**。
platform.vsphere.osDisk.diskSizeGB	以 GB 为单位的磁盘大小。	整数
platform.vsphere.cpus	分配虚拟机的虚拟处理器内核总数。	整数

参数	描述	值
platform.vsphere.coresPerSocket	虚拟机中每个插槽的内核数。虚拟机上的虚拟套接字数量为 **platform.vsphere.cpus/platform.vsphere.coresPerSocket**。默认值为 **1**。	整数
platform.vsphere.memoryMB	以 MB 为单位的虚拟机内存大小。	整数

12.3.10.2. 安装程序置备的 VMware vSphere 集群的 install-config.yaml 文件示例

您可以自定义 install-config.yaml 文件，以指定有关 OpenShift Container Platform 集群平台的更多信息，或修改所需参数的值。

```
apiVersion: v1
baseDomain: example.com (1)
compute: (2)
- hyperthreading: Enabled (3)
  name: worker
  replicas: 3
  platform:
    vsphere: (4)
      cpus: 2
      coresPerSocket: 2
      memoryMB: 8192
      osDisk:
        diskSizeGB: 120
controlPlane: (5)
  hyperthreading: Enabled (6)
  name: master
  replicas: 3
  platform:
    vsphere: (7)
      cpus: 4
      coresPerSocket: 2
      memoryMB: 16384
      osDisk:
        diskSizeGB: 120
metadata:
  name: cluster (8)
networking:
  clusterNetwork:
  - cidr: 10.128.0.0/14
    hostPrefix: 23
  machineNetwork:
  - cidr: 10.0.0.0/16
  networkType: OpenShiftSDN
  serviceNetwork:
  - 172.30.0.0/16
platform:
```

```
  vsphere:
    vcenter: your.vcenter.server
    username: username
    password: password
    datacenter: datacenter
    defaultDatastore: datastore
    folder: folder
    network: VM_Network
    cluster: vsphere_cluster_name 9
    apiVIP: api_vip
    ingressVIP: ingress_vip
fips: false
pullSecret: '{"auths": ...}'
sshKey: 'ssh-ed25519 AAAA...'
```

1 集群的基域。所有 DNS 记录都必须是这个基域的子域，并包含集群名称。

2 5 **controlPlane** 部分是一个单映射，但 **compute** 部分是一系列映射。为满足不同数据结构的要求，**compute** 部分的第一行必须以连字符 **-** 开头，**controlPlane** 部分的第一行则不可以连字符开头。只使用一个 control plane 池。

3 6 是否要启用或禁用并发多线程或**超线程**。默认情况下，启用并发多线程以提高机器内核的性能。您可以通过将参数值设为 **Disabled** 来禁用。如果您在某些集群机器上禁用并发多线程，则必须在所有集群机器上禁用。

重要

如果禁用并发多线程，请确保在容量规划时考虑到机器性能可能会显著降低的问题。如果您禁用并发多线程，则计算机必须至少使用 8 个 CPU 和 32GB RAM。

4 7 可选：为 compute 和 control plane 机器提供额外的机器池参数配置。

8 您在 DNS 记录中指定的集群名称。

9 要在其中安装 OpenShift Container Platform 集群的 vSphere 集群。安装程序使用 vSphere 集群的 root 资源池作为默认资源池。

12.3.10.3. 在安装过程中配置集群范围代理

生产环境可能会拒绝直接访问互联网，而是提供 HTTP 或 HTTPS 代理。您可以通过在 **install-config.yaml** 文件中配置代理设置，将新的 OpenShift Container Platform 集群配置为使用代理。

先决条件

- 您有一个现有的 **install-config.yaml** 文件。
- 您检查了集群需要访问的站点，并决定是否需要绕过代理。默认情况下代理所有集群出口流量，包括对托管云供应商 API 的调用。您需要将站点添加到 **Proxy** 对象的 **spec.noProxy** 字段来绕过代理。

注意

Proxy 对象 **status.noProxy** 字段使用安装配置中的 **networking.machineNetwork[].cidr**、**networking.clusterNetwork[].cidr** 和 **networking.serviceNetwork[]** 字段的值填充。

对于在 Amazon Web Services(AWS)、Google Cloud Platform(GCP)、Microsoft Azure 和 Red Hat OpenStack Platform(RHOSP)上安装, **Proxy** 对象 **status.noProxy** 字段也会使用实例元数据端点填充(**169.254.169.254**)。

流程

1. 编辑 **install-config.yaml** 文件并添加代理设置。例如：

```
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: example.com 3
additionalTrustBundle: | 4
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
...
```

1. 用于创建集群外 HTTP 连接的代理 URL。URL 必须是 **http**。
2. 用于创建集群外 HTTPS 连接的代理 URL。
3. 要排除在代理中的目标域名、IP 地址或其他网络 CIDR 的逗号分隔列表。在域前面加 **.** 来仅匹配子域。例如： **.y.com** 匹配 **x.y.com**，但不匹配 **y.com**。使用 * 绕过所有目的地的代理。您必须包含 vCenter 的 IP 地址以及用于其机器的 IP 范围。
4. 如果提供，安装程序会在 **openshift-config 命名空间中生成名为 user-ca- bundle** 的配置映射来保存额外的 CA 证书。如果您提供 **additionalTrustBundle** 和至少一个代理设置，则 **Proxy** 对象会被配置为引用 **trustedCA** 字段中的 **user-ca-bundle** 配置映射。然后，Cluster Network Operator 会创建一个 **trusted-ca-bundle** 配置映射，该配置映射将为 **trustedCA** 参数指定的内容与 RHCOS 信任捆绑包合并。 **additionalTrustBundle** 字段是必需的，除非代理的身份证书由来自 RHCOS 信任捆绑包的颁发机构签名。

注意

安装程序不支持代理的 **readinessEndpoints** 字段。

2. 保存该文件，并在安装 OpenShift Container Platform 时引用。

安装程序会创建一个名为 **cluster** 的集群范围代理，该代理使用提供的 **install-config.yaml** 文件中的代理设置。如果没有提供代理设置，仍然会创建一个 **cluster Proxy** 对象，但它会有一个空 **spec**。

注意

只支持名为 **cluster** 的 **Proxy** 对象，且无法创建额外的代理。

12.3.11. 网络配置阶段

当在安装前指定集群配置时，在安装过程中的几个阶段可以修改网络配置：

阶段 1

输入 **openshift-install create install-config** 命令后。在 **install-config.yaml** 文件中，您可以自定义以下与网络相关的字段：

- **networking.networkType**
- **networking.clusterNetwork**
- **networking.serviceNetwork**
- **networking.machineNetwork**
  有关这些字段的更多信息，请参阅"安装配置参数"。

注意

将 **networking.machineNetwork** 设置为与首选 NIC 所在的 CIDR 匹配。

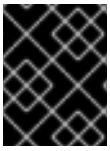
阶段 2

输入 **openshift-install create manifests** 命令后。如果必须指定高级网络配置，在这个阶段中，只能使用您要修改的字段来定义自定义的 Cluster Network Operator 清单。

在 2 阶段，您无法覆盖 **install-config.yaml** 文件中的 1 阶段中指定的值。但是，您可以在第 2 阶段进一步自定义集群网络供应商。

12.3.12. 指定高级网络配置

您可以通过为集群网络供应商指定额外的配置，使用高级配置自定义将集群整合到现有网络环境中。您只能在安装集群前指定高级网络配置。



重要

不支持修改安装程序创建的 OpenShift Container Platform 清单文件。支持应用您创建的清单文件，如以下流程所示。

先决条件

- 创建 **install-config.yaml** 文件并完成对其所做的任何修改。

流程

1. 进入包含安装程序的目录并创建清单：

```
$ ./openshift-install create manifests --dir <installation_directory>
```

其中：

<installation_directory>
指定包含集群的 **install-config.yaml** 文件的目录名称。

2. 在 **<installation_directory>/manifests/** 目录下，为高级网络配置创建一个名为 **cluster-network-03-config.yml** 的 stub 清单文件：

```
$ cat <<EOF > <installation_directory>/manifests/cluster-network-03-config.yml
apiVersion: operator.openshift.io/v1
kind: Network
metadata:
  name: cluster
spec:
EOF
```

其中：

<installation_directory>

指定包含集群的 **manifests/** 目录的目录名称。

3. 在编辑器中打开 **cluster-network-03-config.yml** 文件，并为集群指定高级网络配置，如下例所示：

为 OpenShift SDN 网络供应商指定不同的 VXLAN 端口

```
apiVersion: operator.openshift.io/v1
kind: Network
metadata:
  name: cluster
spec:
  defaultNetwork:
    openshiftSDNConfig:
      vxlanPort: 4800
```

4. 保存 **cluster-network-03-config.yml** 文件，再退出文本编辑器。

5. 可选：备份 **manifests/cluster-network-03-config.yml** 文件。创建集群时，安装程序会删除 **manifests/** 目录。

12.3.13. Cluster Network Operator 配置

集群网络的配置作为 Cluster Network Operator (CNO) 配置的一部分被指定，并存储在名为 **cluster**的自定义资源（CR）对象中。CR 指定 **operator.openshift.io** API 组中的 **Network** API 的字段。

CNO 配置会在集群安装过程中从 **Network.config.openshift.io** API 组中的 **Network** API 继承以下字段，这些字段无法更改：

clusterNetwork

从中分配 pod IP 地址的 IP 地址池。

serviceNetwork

服务的 IP 地址池。

defaultNetwork.type

集群网络供应商，如 OpenShift SDN 或 OVN-Kubernetes。

您可以通过在名为 **cluster** 的 CNO 对象中设置 **defaultNetwork** 对象的字段来为集群指定集群网络供应商配置。

12.3.13.1. Cluster Network Operator 配置对象

Cluster Network Operator（CNO）的字段在下表中描述：

表 12.26. Cluster Network Operator 配置对象

字段	类型	描述
metadata.name	**字符串**	CNO 对象的名称。这个名称始终是 **cluster**。

spec.clusterNetwork	**数组**	用于指定从哪些 IP 地址块分配 Pod IP 地址以及分配给集群中每个节点的子网前缀长度的列表。例如：
		`spec:`
		`  clusterNetwork:`
		`  - cidr: 10.128.0.0/19`
		`  hostPrefix: 23`
		`  - cidr: 10.128.32.0/19`
		`  hostPrefix: 23`
		此值是只读的，并在 **install-config.yaml** 文件中指定。

spec.serviceNetwork	**数组**	服务的 IP 地址块。OpenShift SDN 和 OVN-Kubernetes Container Network Interface（CNI）网络供应商只支持服务网络具有单个 IP 地址块。例如：
		`spec:`
		`  serviceNetwork:`
		`  - 172.30.0.0/14`
		此值是只读的，并在 **install-config.yaml** 文件中指定。

spec.defaultNetwork	**对象**	为集群网络配置 Container Network Interface（CNI）集群网络供应商。
spec.kubeProxyConfig	**对象**	此对象的字段指定 kube-proxy 配置。如果您使用 OVN-Kubernetes 集群网络供应商，则 kube-proxy 的配置不会起作用。

defaultNetwork 对象配置
defaultNetwork 对象的值在下表中定义：

表 12.27. defaultNetwork 对象

字段	类型	描述

字段	类型	描述
type	**字符串**	**OpenShiftSDN** 或 **OVNKubernetes**。在安装过程中选择了集群网络供应商。集群安装后无法更改这个值。  **注意** OpenShift Container Platform 默认使用 OpenShift SDN Container Network Interface（CNI）集群网络供应商。
openshiftSDNConfig	**对象**	此对象仅对 OpenShift SDN 集群网络供应商有效。
ovnKubernetesConfig	**对象**	此对象仅对 OVN-Kubernetes 集群网络供应商有效。

配置 OpenShift SDN CNI 集群网络供应商

下表描述了 OpenShift SDN Container Network Interface（CNI）集群网络供应商的配置字段。

表 12.28. openshiftSDNConfig 对象

字段	类型	描述
mode	**字符串**	配置 OpenShift SDN 的网络隔离模式。默认值为 **NetworkPolicy**。 **Multitenant** 和 **Subnet** 的值可以向后兼容 OpenShift Container Platform 3.x，但不推荐这样做。集群安装后无法更改这个值。
mtu	**整数**	VXLAN 覆盖网络的最大传输单元 (MTU) 。这根据主网络接口的 MTU 自动探测。您通常不需要覆盖检测到的 MTU。 如果自动探测的值不是您期望的，请确认节点上主网络接口中的 MTU 是正确的。您不能使用这个选项更改节点上主网络接口的 MTU 值。 如果您的集群中的不同节点需要不同的 MTU 值，则必须将此值设置为比集群中的最低 MTU 值小 **50**。例如，如果集群中的某些节点的 MTU 为 **9001**，而某些节点的 MTU 为 **1500**，则必须将此值设置为 **1450**。 集群安装后无法更改这个值。

字段	类型	描述
vxlanPort	**整数**	用于所有 VXLAN 数据包的端口。默认值为 **4789**。集群安装后无法更改这个值。 如果您在虚拟环境中运行，并且现有节点是另一个 VXLAN 网络的一部分，那么可能需要更改此值。例如，当在 VMware NSX-T 上运行 OpenShift SDN 覆盖时，您必须为 VXLAN 选择一个备用端口，因为两个 SDN 都使用相同的默认 VXLAN 端口号。 在 Amazon Web Services (AWS) 上，您可以在端口 **9000** 和端口 **9999** 之间为 VXLAN 选择一个备用端口。

OpenShift SDN 配置示例

```
defaultNetwork:
  type: OpenShiftSDN
  openshiftSDNConfig:
    mode: NetworkPolicy
    mtu: 1450
    vxlanPort: 4789
```

配置 OVN-Kubernetes CNI 集群网络供应商

下表描述了 OVN-Kubernetes CNI 集群网络供应商的配置字段。

表 12.29. ovnKubernetesConfig 对象

字段	类型	描述
mtu	**整数**	Geneve（Generic Network Virtualization Encapsulation）覆盖网络的最大传输单元（MTU）。这根据主网络接口的 MTU 自动探测。您通常不需要覆盖检测到的 MTU。 如果自动探测的值不是您期望的，请确认节点上主网络接口中的 MTU 是正确的。您不能使用这个选项更改节点上主网络接口的 MTU 值。 如果您的集群中的不同节点需要不同的 MTU 值，则必须将此值设置为比集群中的最低 MTU 值小 **100**。例如，如果集群中的某些节点的 MTU 为 **9001**，而某些节点的 MTU 为 **1500**，则必须将此值设置为 **1400**。 集群安装后无法更改这个值。
genevePort	**整数**	用于所有 Geneve 数据包的端口。默认值为 **6081**。集群安装后无法更改这个值。

OVN-Kubernetes 配置示例

```
defaultNetwork:
  type: OVNKubernetes
```

```
ovnKubernetesConfig:
  mtu: 1400
  genevePort: 6081
```

kubeProxyConfig 对象配置
kubeProxyConfig 对象的值在下表中定义：

表 12.30. kubeProxyConfig 对象

字段	类型	描述
iptablesSyncPeriod	**字符串**	**iptables** 规则的刷新周期。默认值为 **30s**。有效的后缀包括 **s**、**m** 和 **h**，具体参见 Go **time** 软件包文档。 **注意** 由于 OpenShift Container Platform 4.3 及更高版本中引进了性能上的改进，现在不再需要调整 **iptablesSyncPeriod** 参数。

proxyArguments.iptables-min-sync-period	**数组**	刷新 **iptables** 规则前的最短时长。此字段确保刷新的频率不会过于频繁。有效的后缀包括 **s**、**m** 和 **h**，具体参见 Go **time** 软件包。默认值为：
		
		`kubeProxyConfig:`
		`  proxyArguments:`
		`  iptables-min-sync-period:`
		`  - 0s`

12.3.14. 部署集群

您可以在兼容云平台中安装 OpenShift Container Platform。

重要

安装程序的 **create cluster** 命令只能在初始安装过程中运行一次。

先决条件

- 配置托管集群的云平台的帐户。
- 获取 OpenShift Container Platform 安装程序以及集群的 pull secret。

流程

1. 更改为包含安装程序的目录并初始化集群部署：

```
$ ./openshift-install create cluster --dir <installation_directory> \ 1
    --log-level=info 2
```

1. 对于 **<installation_directory>**，请指定自定义 **./install-config.yaml** 文件的位置。
2. 要查看不同的安装详情，请指定 **warn**、**debug** 或 **error**，而不要指定 **info**。

重要

使用托管在 VMC 环境中的堡垒中的 **openshift-install** 命令。

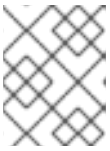
注意

如果您在主机上配置的云供应商帐户没有足够的权限来部署集群，安装过程将会停止，并且显示缺少的权限。

集群部署完成后，终端会显示访问集群的信息，包括指向其 Web 控制台的链接和 **kubeadmin** 用户的凭证。

输出示例

```
...
INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run 'export
KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'
INFO Access the OpenShift web-console here: https://console-openshift-
console.apps.mycluster.example.com
INFO Login to the console with user: "kubeadmin", and password: "4vYBz-Ee6gm-ymBZj-
Wt5AL"
INFO Time elapsed: 36m22s
```



注意

当安装成功时，集群访问和凭证信息还会输出到 **<installation_directory>/.openshift_install.log**。

重要

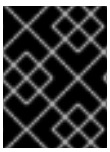
- 安装程序生成的 Ignition 配置文件包含在 24 小时后过期的证书，然后在过期时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外情况是，您需要手动批准待处理的 **node-bootstrapper** 证书签名请求（CSR）来恢复 kubelet 证书。如需更多信息，请参阅*从过期的 control plane 证书中恢复*的文档。
- 建议您在生成 12 小时后使用 Ignition 配置文件，因为集群安装后 24 小时证书从 16 小时轮转至 22 小时。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中运行证书更新时避免安装失败。

重要

您不得删除安装程序或安装程序所创建的文件。需要这两者才能删除集群。

12.3.15. 通过下载二进制文件安装 OpenShift CLI

您需要安装 CLI（**oc**） 来使用命令行界面与 OpenShift Container Platform 进行交互。您可在 Linux 、Windows 或 macOS 上安装 **oc**。



重要

如果安装了旧版本的 **oc**，则无法使用 OpenShift Container Platform 4.6 中的所有命令。下载并安装新版本的 **oc**。

12.3.15.1. 在 Linux 上安装 OpenShift CLI

您可以按照以下流程在 Linux 上安装 OpenShift CLI（**oc**）二进制文件。

流程

1. 进入到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 在 **Version** 下拉菜单中选择相应的版本。
3. 单击 **OpenShift v4.6 Linux 客户端** 条目旁边的 **Download Now**，再保存文件。
4. 解包存档：

```
$ tar xvzf <file>
```

5. 把 **oc** 二进制代码放到 **PATH** 中的目录中。
   执行以下命令可以查看当前的 **PATH** 设置：

```
$ echo $PATH
```

安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
$ oc <command>
```

12.3.15.2. 在 Windows 上安装 OpenShift CLI

您可以按照以下流程在 Windows 上安装 OpenShift CLI（**oc**）二进制代码。

流程

1. 进入到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 在 **Version** 下拉菜单中选择相应的版本。
3. 单击 **OpenShift v4.6 Windows 客户端**条目旁边的 **Download Now**，再保存文件。
4. 使用 ZIP 程序解压存档。
5. 把 **oc** 二进制代码放到 **PATH** 中的目录中。
   要查看您的 **PATH**，请打开命令提示窗口并执行以下命令：

```
C:\> path
```

安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
C:\> oc <command>
```

12.3.15.3. 在 macOS 上安装 OpenShift CLI

您可以按照以下流程在 macOS 上安装 OpenShift CLI（**oc**）二进制代码。

流程

1. 进入到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 在 **Version** 下拉菜单中选择相应的版本。
3. 单击 **OpenShift v4.6 MacOSX 客户端**条目旁边的 **Download Now**，再保存文件。
4. 解包和解压存档。
5. 将 **oc** 二进制文件移到 PATH 的目录中。
   要查看您的 **PATH**，打开一个终端窗口并执行以下命令：

   ```
   $ echo $PATH
   ```

安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
$ oc <command>
```

12.3.16. 使用 CLI 登录到集群

您可以通过导出集群 **kubeconfig** 文件，以默认系统用户身份登录集群。**kubeconfig** 文件包含关于集群的信息，供 CLI 用于将客户端连接到正确集群和 API 服务器。该文件特只适用于一个特定的集群，在 OpenShift Container Platform 安装过程中创建。

先决条件

- 已部署了 OpenShift Container Platform 集群。
- 已安装 **oc** CLI。

流程

1. 导出 **kubeadmin** 凭证：

   ```
   $ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
   ```

   1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

2. 使用导出的配置，验证能否成功运行 **oc** 命令：

   ```
   $ oc whoami
   ```

   输出示例

```
system:admin
```

12.3.17. 创建 registry 存储

安装集群后，必须为 registry Operator 创建存储。

12.3.17.1. 安装过程中删除的镜像 registry

在不提供可共享对象存储的平台上，OpenShift Image Registry Operator bootstraps 本身的状态是 **Removed**。这允许 **openshift-installer** 在这些平台类型上完成安装。

将 **ManagementState** Image Registry Operator 配置从 **Removed** 改为 **Managed**。

注意

Prometheus 控制台提供了一个 **ImageRegistryRemoved** 警报，例如：

"Image Registry has been removed.**ImageStreamTags**, **BuildConfigs** and **DeploymentConfigs** which reference **ImageStreamTags** may not work as expected.Please configure storage and update the config to **Managed** state by editing configs.imageregistry.operator.openshift.io."

12.3.17.2. 镜像 registry 存储配置

对于不提供默认存储的平台，Image Registry Operator 最初将不可用。安装后，您必须配置 registry 使用的存储，这样 Registry Operator 才可用。

示配置生产集群所需的持久性卷的说明。如果适用，显示有关将空目录配置为存储位置的说明，该位置只可用于非生产集群。

另外还提供了在升级过程中使用 **Recreate** rollout 策略来允许镜像 registry 使用块存储类型的说明。

12.3.17.2.1. 为 VMware vSphere 配置 registry 存储

作为集群管理员，在安装后需要配置 registry 来使用存储。

先决条件

- 具有 Cluster Administrator 权限
- VMware vSphere上有一个集群。
- 为集群置备的持久性存储，如 Red Hat OpenShift Container Storage。

重要

如果您只有一个副本，OpenShift Container Platform 支持对镜像 registry 存储的 **ReadWriteOnce** 访问。要部署支持高可用性的、带有两个或多个副本的镜像 registry，需要 **ReadWriteMany** 访问设置。

- 必须有“ 100Gi”容量。

重要

测试显示，在 RHEL 中使用 NFS 服务器作为核心服务的存储后端可能会出现问题。这包括 OpenShift Container Registry 和 Quay，Prometheus 用于监控存储，以及 Elasticsearch 用于日志存储。因此，不推荐使用 RHEL NFS 作为 PV 后端用于核心服务。

市场上的其他 NFS 实现可能没有这些问题。如需了解更多与此问题相关的信息，请联络相关的 NFS 厂商。

流程

1. 为了配置 registry 使用存储，需要修改 **configs.imageregistry/cluster** 资源中的 **spec.storage.pvc** 。

注意

使用共享存储时，请查看您的安全设置以防止被外部访问。

2. 验证您没有 registry pod:

```
$ oc get pod -n openshift-image-registry
```

注意

如果存储类型为 **emptyDIR**，则副本数不能超过 **1**。

3. 检查 registry 配置：

```
$ oc edit configs.imageregistry.operator.openshift.io
```

输出示例

```
storage:
  pvc:
    claim: 1
```

1 将 **claim** 字段留空以允许自动创建一个 **image-registry-storage** PVC。

4. 检查 **clusteroperator** 的状态：

```
$ oc get clusteroperator image-registry
```

12.3.17.2.2. 为 VMware vSphere 配置块 registry 存储

在作为集群管理员升级时，要允许镜像 registry 使用块存储类型，如 vSphere Virtual Machine Disk（VMDK），您可以使用 **Recreate** rollout 策略。

重要

支持块存储卷，但不建议将其用于生产环境中的镜像 registry。在块存储上配置 registry 的安装不具有高可用性，因为 registry 无法拥有多个副本。

流程

1. 要将镜像 registry 存储设置为块存储类型，对 registry 进行补丁，使其使用 **Recreate** rollout 策略，且仅使用 **1** 个副本运行：

```
$ oc patch config.imageregistry.operator.openshift.io/cluster --type=merge -p '{"spec":
{"rolloutStrategy":"Recreate","replicas":1}}'
```

2. 为块存储设备置备 PV，并为该卷创建 PVC。请求的块卷使用 ReadWriteOnce（RWO）访问模式。

   a. 创建包含以下内容的 **pvc.yaml** 文件以定义 VMware vSphere **PersistentVolumeClaim**：

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: image-registry-storage 1
  namespace: openshift-image-registry 2
spec:
  accessModes:
  - ReadWriteOnce 3
  resources:
    requests:
      storage: 100Gi 4
```

   1. 代表 **PersistentVolumeClaim** 对象的唯一名称。
   2. **PersistentVolumeClaim** 对象的命名空间，即 **openshift-image-registry**。
   3. 持久性卷声明的访问模式。使用 **ReadWriteOnce** 时，单个节点可以通过读写权限挂载这个卷。
   4. 持久性卷声明的大小。

   b. 从文件创建 **PersistentVolumeClaim** 对象：

```
$ oc create -f pvc.yaml -n openshift-image-registry
```

3. 编辑 registry 配置，使其可以正确引用 PVC:

```
$ oc edit config.imageregistry.operator.openshift.io -o yaml
```

输出示例

```
storage:
  pvc:
    claim: 1
```

1. 通过创建自定义 PVC，您可以将 **claim** 字段留空以用于默认自动创建 **image-registry-storage** PVC。

有关配置 registry 存储以便引用正确的 PVC 的说明，请参阅为 vSphere 配置 registry。

12.3.18. 备份 VMware vSphere 卷

OpenShift Container Platform 将新卷作为独立持久性磁盘置备，以便在集群中的任何节点上自由附加和分离卷。因此，无法备份使用快照的卷，也无法从快照中恢复卷。如需更多信息，请参阅 快照限制。

流程

要创建持久性卷的备份：

1. 停止使用持久性卷的应用程序。
2. 克隆持久性卷。
3. 重启应用程序。
4. 创建克隆的卷的备份。
5. 删除克隆的卷。

12.3.19. OpenShift Container Platform 的 Telemetry 访问

在 OpenShift Container Platform 4.6 中，默认运行的 Telemetry 服务提供有关集群健康状况和成功更新的指标，需要访问互联网。如果您的集群连接到互联网，Telemetry 会自动运行，而且集群会注册到 OpenShift Cluster Manager。

确认 OpenShift Cluster Manager 清单正确后，可以由 Telemetry 自动维护，也可以使用 OpenShift Cluster Manager 手动维护，使用订阅监控来跟踪帐户或多集群级别的 OpenShift Container Platform 订阅。

其他资源

- 有关 Telemetry 服务的更多信息，请参阅关于远程健康监控。

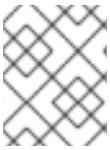
12.3.20. 后续步骤

- 自定义集群。
- 如果需要，您可以选择不使用远程健康报告。
- 设置 registry 并配置 registry 存储。

12.4. 在受限网络中的 VMC 上安装集群

在 OpenShift Container Platform 版本 4.6 中，您可以通过将其部署到 VMware Cloud（VMC）on AWS 来在受限网络的 VMware vSphere 基础架构上安装集群。

为 OpenShift Container Platform 部署配置了 VMC 环境后，您要使用堡垒管理主机中的 OpenShift Container Platform 安装程序，并位于 VMC 环境中。安装程序和 control plane 可以自动部署和管理 OpenShift Container Platform 集群所需的资源。

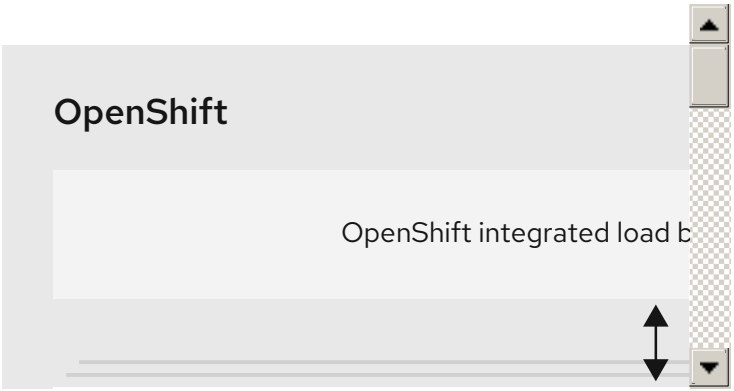


注意

OpenShift Container Platform 支持将集群部署到单个 VMware vCenter 中。不支持在多个 vCenter 上使用机器/机器集部署集群。

12.4.1. 为 vSphere 设置 VMC

您可以在 AWS 托管的 vSphere 集群上安装 OpenShift Container Platform（VMC），以便启用在混合云内部和内部管理应用程序。



在 VMware vSphere 上安装 OpenShift Container Platform 之前，您必须在 VMC 环境中配置多个选项。确定您的 VMC 环境有以下先决条件：

- 创建非专用的、启用了 DHCP 的、NSX-T 网络片段和子网。其他虚拟机（VM）可以托管在子网上，但至少有 8 个 IP 地址可用于 OpenShift Container Platform 部署。
- 在 DHCP 范围之外分配两个 IP 地址，并使用反向 DNS 记录配置它们。
  - **api.<cluster_name>.<base_domain>** 的 DNS 记录，指向分配的 IP 地址。
  - ***.apps.<cluster_name>.<base_domain>** 的 DNS 记录，指向分配的 IP 地址。
- 配置以下防火墙规则：
  - 在安装主机和软件定义的数据中心（SDDC）管理网络之间的端口 443 的 ANY:ANY 防火墙规则。这可让您在部署过程中上传 Red Hat Enterprise Linux CoreOS（RHCOS）OVA。
  - OpenShift Container Platform 计算网络和 vCenter 间的 HTTPS 防火墙规则。此连接允许 OpenShift Container Platform 与 vCenter 通信，以置备和管理节点、持久性卷声明（PVC）和其他资源。
- 您必须具有以下信息才能部署 OpenShift Container Platform:
  - OpenShift Container Platform 集群名称，如 **vmc-prod-1**。
  - 基本 DNS 名称，如 **companyname.com**。
  - 如果没有使用默认设置，则必须识别 Pod 网络 CIDR 和服务网络 CIDR，默认设置为 **10.128.0.0/14** 和 **172.30.0.0/16**。这些 CIDR 用于 pod 到 pod 和 pod 到服务的通信，且无法在外部访问，但不得与机构中的现有子网重叠。
  - 以下 vCenter 信息：
    - vCenter 主机名、用户名和密码
    - 数据中心名称，如 **SDDC-Datacenter**
    - 集群名称，如 **Cluster-1**
    - 网络名称

- 数据存储名称，如 **WorkloadDatastore**

注意

建议在集群安装完成后将 vSphere 集群移到 VMC **Compute-ResourcePool** 资源池。

- 基于 Linux 的主机作为堡垒部署到 VMC。
  - 堡垒主机可以是 Red Hat Enterprise Linux（RHEL）或其他基于 Linux 的主机，它必须具有互联网连接，并可以将 OVA 上传到 ESXi 主机。
  - 将 OpenShift CLI 工具下载并安装到堡垒主机。
    - **openshift-install** 安装程序
    - OpenShift CLI（**oc**）工具

注意

您不能将 VMware NSX Container Plugin 用于 Kubernetes（NCP），NCP 不使用 NSX 作为 OpenShift SDN。目前 VMC 提供的 NSX 版本与 OpenShift Container Platform 认证的 NCP 版本不兼容。

但是，NSX DHCP 服务会通过全堆栈自动 OpenShift Container Platform 部署来管理虚拟机 IP 管理，节点可以手动或自动置备，可通过 Machine API 与 vSphere 集成。另外，还创建 NSX 防火墙规则，以便启用 OpenShift Container Platform 集群以及堡垒主机和 VMC vSphere 主机间的访问。

12.4.1.1. VMC Sizer 工具

AWS 上的 VMware Cloud 基于 AWS 裸机基础架构构建，这与运行 AWS 原生服务的裸机基础架构相同。当在 AWS 软件定义的数据中心（SDDC）上部署 VMware 云时，您要将这些物理服务器节点以单一租户方式运行 VMware ESXi hypervisor。这意味着其他使用 VMC 的用户无法访问物理基础结构。务必要考虑托管您的虚拟基础架构所需的物理主机数量。

要确定这一点，VMware 在 AWS Sizer 上提供 VMC。使用这个工具，您可以定义要在 VMC 上托管的资源：

- 工作负载类型
- 虚拟机总数
- 规格信息，例如：
  - 存储要求
  - vCPUs
  - vRAM
  - 过量使用比例

通过这些信息，sizer 工具可以根据 VMware 最佳实践生成报告，并推荐集群配置和您需要的主机数量。

12.4.2. vSphere 先决条件

- 在镜像主机上创建镜像 registry，并获取您的 OpenShift Container Platform 版本的 **imageContentSources** 数据。

重要

由于安装介质位于堡垒主机上，因此请使用该计算机完成所有安装步骤。

- 置备块 registry 存储。如需有关持久性存储的更多信息，请参阅了解持久性存储。
- 查看有关 OpenShift Container Platform 安装和更新流程的详细信息。
- 如果使用防火墙并计划使用遥测（telemetry），您必须将防火墙配置为允许集群需要访问的站点。

注意

如果要配置代理，请务必还要查看此站点列表。

12.4.3. 关于在受限网络中安装

在 OpenShift Container Platform 4.6 中，可以执行不需要有效的互联网连接来获取软件组件的安装。受限网络安装可使用安装程序置备的基础架构或用户置备的基础架构完成，具体取决于您要安装集群的云平台。

如果选择在云平台中执行受限网络安装，仍然需要访问其云 API。有些云功能，比如 Amazon Web Service 的 Route 53 DNS 和 IAM 服务，需要访问互联网。根据您的网络，在裸机硬件或 VMware vSphere 上安装时可能需要较少的互联网访问。

要完成受限网络安装，您必须创建一个 registry，镜像 OpenShift Container Platform registry 的内容并包含其安装介质。您可以在堡垒主机上创建此镜像，该主机可同时访问互联网和您的封闭网络，也可以使用满足您的限制条件的其他方法。

12.4.3.1. 其他限制

受限网络中的集群还有以下额外限制：

- **ClusterVersion** 状态包含一个 **Unable to retrieve available updates** 错误。
- 默认情况下，您无法使用 Developer Catalog 的内容，因为您无法访问所需的镜像流标签。

12.4.4. OpenShift Container Platform 的互联网访问

在 OpenShift Container Platform 4.6 中，您需要访问互联网来获得用来安装集群的镜像。

您必须具有以下互联网访问权限：

- 访问 OpenShift Cluster Manager 以下载安装程序并执行订阅管理。如果集群可以访问互联网，并且没有禁用 Telemetry，该服务会自动授权您的集群。
- 访问 Quay.io，以获取安装集群所需的软件包。
- 获取执行集群更新所需的软件包。

重要

如果您的集群无法直接访问互联网，则可以在置备的某些类基础架构上执行受限网络安装。在此过程中，您要下载所需的内容，并使用它在镜像 registry（mirror registry） 中填充安装集群并生成安装程序所需的软件包。对于某些安装类型，集群要安装到的环境不需要访问互联网。在更新集群之前，要更新 registry 镜像系统中的内容。

12.4.5. VMware vSphere 基础架构要求

您必须在满足您使用的组件要求的 VMware vSphere 版本 6 或 7 实例上安装 OpenShift Container Platform 集群。

表 12.31. VMware 组件支持的最低 vSphere 版本

组件	最低支持版本	描述
虚拟机监控程序	vSphere 6.5 及之后的版本 13	此版本是 Red Hat Enterprise Linux CoreOS(RHCOS)支持的最低版本。请查看 Red Hat Enterprise Linux 8 支持的管理程序列表。
使用 in-tree 驱动程序存储	vSphere 6.5 及之后的版本	此插件使用 OpenShift Container Platform 中包含的 vSphere 的树内存储驱动程序创建 vSphere 存储。

如果您使用 vSphere 版本 6.5 实例，请在安装 OpenShift Container Platform 前考虑升级到 6.7U3 或 7.0。

重要

您必须确保在安装 OpenShift Container Platform 前同步 ESXi 主机上的时间。请参阅 VMware 文档中的编辑主机时间配置。

12.4.6. 网络连接要求

您必须配置机器之间的网络连接，以允许 OpenShift Container Platform 集群组件进行通信。

查看有关所需网络端口的以下详细信息。

表 12.32. 用于全机器到所有机器通信的端口

协议	端口	描述
ICMP	N/A	网络可访问性测试
TCP	**1936**	指标
	9000-9999	主机级别的服务，包括端口 **9100-9101** 上的节点导出器和端口 **9099** 上的 Cluster Version Operator。

协议	端口	描述
	10250-10259	Kubernetes 保留的默认端口
	10256	openshift-sdn
UDP	**4789**	虚拟可扩展 LAN(VXLAN)
	6081	Geneve
	9000-9999	主机级别的服务，包括端口 **9100-9101** 上的节点导出器。
	500	IPsec IKE 数据包
	4500	IPsec NAT-T 数据包
TCP/UDP	**30000-32767**	Kubernetes 节点端口
ESP	N/A	IPsec Encapsulating Security Payload(ESP)

表 12.33. 用于所有机器控制平面通信的端口

协议	端口	描述
TCP	**6443**	Kubernetes API

表 12.34. control plane 机器用于 control plane 机器通信的端口

协议	端口	描述
TCP	**2379-2380**	etcd 服务器和对等端口

12.4.7. vCenter 要求

在使用安装程序置备的基础架构的 vCenter 上安装 OpenShift Container Platform 集群前，您必须准备自己的环境。

所需的 vCenter 帐户权限
要在 vCenter 中安装 OpenShift Container Platform 集群，安装程序需要一个具有特权的帐户来读取和创建所需资源。使用具有全局管理特权的帐户是访问所有必要权限的最简单方式。

如果无法使用具有全局管理特权的帐户，您必须创建角色来授予 OpenShift Container Platform 集群安装所需的权限。虽然大多数权限始终是必需的，但是一些权限只有在计划安装程序需要在您的 vCenter 实例中置备一个包含 OpenShift Container Platform 集群的文件夹时（这是默认行为）才需要。您必须为指定对象创建或修改 vSphere 角色，才能授予所需的权限。

如果安装程序创建 vSphere 虚拟机文件夹，则需要额外的角色。

例 12.7. 安装所需的角色和权限

角色的 vSphere 对象	何时需要	所需的权限
vSphere vCenter	Always	**Cns.Searchable** **InventoryService.Tagging.AttachTag** **InventoryService.Tagging.CreateCategory** **InventoryService.Tagging.CreateTag** **InventoryService.Tagging.DeleteCategory** **InventoryService.Tagging.DeleteTag** **InventoryService.Tagging.EditCategory** **InventoryService.Tagging.EditTag** **Sessions.ValidateSession** **StorageProfile.View**
vSphere vCenter Cluster	Always	**Host.Config.Storage** **Resource.AssignVMToPool** **VApp.AssignResourcePool** **VApp.Import** **VirtualMachine.Config.AddNewDisk**
vSphere Datastore	Always	**Datastore.AllocateSpace** **Datastore.Browse** **Datastore.FileManagement**
vSphere 端口组	Always	**Network.Assign**

角色的 vSphere 对象	何时需要	所需的权限
虚拟机文件夹	Always	**Resource.AssignVMToPool** **VApp.Import** **VirtualMachine.Config.Add ExistingDisk** **VirtualMachine.Config.Add NewDisk** **VirtualMachine.Config.Add RemoveDevice** **VirtualMachine.Config.Adva ncedConfig** **VirtualMachine.Config.Anno tation** **VirtualMachine.Config.CPU Count** **VirtualMachine.Config.Disk Extend** **VirtualMachine.Config.Disk Lease** **VirtualMachine.Config.Edit Device** **VirtualMachine.Config.Mem ory** **VirtualMachine.Config.Rem oveDisk** **VirtualMachine.Config.Rena me** **VirtualMachine.Config.Rese tGuestInfo** **VirtualMachine.Config.Reso urce** **VirtualMachine.Config.Setti ngs** **VirtualMachine.Config.Upgr adeVirtualHardware** **VirtualMachine.Interact.Gue stControl** **VirtualMachine.Interact.Pow erOff** **VirtualMachine.Interact.Pow erOn** **VirtualMachine.Interact.Res et** **VirtualMachine.Inventory.Cr eate** **VirtualMachine.Inventory.Cr eateFromExisting** **VirtualMachine.Inventory.D elete** **VirtualMachine.Provisionin g.Clone**

角色的 vSphere 对象	何时需要	所需的权限
vSphere vCenter Datacenter	如果安装程序创建虚拟机文件夹	**Resource.AssignVMToPool VApp.Import VirtualMachine.Config.Add ExistingDisk VirtualMachine.Config.Add NewDisk VirtualMachine.Config.Add RemoveDevice VirtualMachine.Config.Adva ncedConfig VirtualMachine.Config.Anno tation VirtualMachine.Config.CPU Count VirtualMachine.Config.Disk Extend VirtualMachine.Config.Disk Lease VirtualMachine.Config.Edit Device VirtualMachine.Config.Mem ory VirtualMachine.Config.Rem oveDisk VirtualMachine.Config.Rena me VirtualMachine.Config.Rese tGuestInfo VirtualMachine.Config.Reso urce VirtualMachine.Config.Setti ngs VirtualMachine.Config.Upgr adeVirtualHardware VirtualMachine.Interact.Gue stControl VirtualMachine.Interact.Pow erOff VirtualMachine.Interact.Pow erOn VirtualMachine.Interact.Res et VirtualMachine.Inventory.Cr eate VirtualMachine.Inventory.Cr eateFromExisting VirtualMachine.Inventory.D elete VirtualMachine.Provisionin g.Clone Folder.Create Folder.Delete**

此外，用户需要一些 **ReadOnly** 权限，某些角色需要权限来提升对子对象的权限。这些设置会根据您是否将集群安装到现有文件夹而有所不同。

例 12.8. 所需的权限和传播设置

vSphere 对象	文件夹类型	传播到子对象	所需的权限
vSphere vCenter	Always	False	列出所需的权限
vSphere vCenter Datacenter	现有文件夹	False	**ReadOnly** 权限
	安装程序创建文件夹	True	列出所需的权限
vSphere vCenter Cluster	Always	True	列出所需的权限
vSphere vCenter Datastore	Always	False	列出所需的权限
vSphere Switch	Always	False	**ReadOnly** 权限
vSphere 端口组	Always	False	列出所需的权限
vSphere vCenter Virtual Machine Folder	现有文件夹	True	列出所需的权限

有关只使用所需权限创建帐户的更多信息，请参阅 vSphere 文档中的 vSphere 权限和用户管理任务。

将 OpenShift Container Platform 与 vMotion 搭配使用

如果要在 vSphere 环境中使用 vMotion，请在安装 OpenShift Container Platform 集群前考虑以下内容。

- OpenShift Container Platform 通常支持仅用于计算的 vMotion。使用 Storage vMotion 可能会导致问题且不被支持。
  为了帮助确保计算和 control plane 节点的正常运行时间，建议您遵循 VMware 最佳实践进行 vMotion。还建议使用 VMware 反关联性规则来改进 OpenShift Container Platform 在维护或硬件问题期间的可用性。

  有关 vMotion 和 anti-affinity 规则的更多信息，请参阅 VMware vSphere 文档 了解 vMotion 网络要求和虚拟机反关联性规则。

- 如果您在 pod 中使用 vSphere 卷，请手动或通过 Storage vMotion 在数据存储间迁移虚拟机，从而导致 OpenShift Container Platform 持久性卷(PV)对象中的无效引用。这些引用可防止受影响的 pod 启动，并可能导致数据丢失。

- 同样，OpenShift Container Platform 不支持在数据存储间有选择地迁移 VMDK、使用数据存储集群进行虚拟机置备、动态或静态置备 PV，或使用作为数据存储集群一部分的数据存储进行 PV 的动态或静态置备。

集群资源

当部署使用安装程序置备的基础架构的 OpenShift Container Platform 集群时，安装程序必须能够在 vCenter 实例中创建多个资源。

标准 OpenShift Container Platform 安装会创建以下 vCenter 资源：

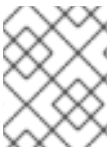
- 1 个文件夹
- 1 标签（Tag）类别
- 1 个标签（Tag）
- 虚拟机:
  - 1 个模板
  - 1 个临时 bootstrap 节点
  - 3 个 control plane 节点
  - 3 个计算机器

虽然这些资源使用了 856 GB 存储，但 bootstrap 节点会在集群安装过程中被销毁。使用标准集群至少需要 800 GB 存储。

如果部署了更多计算机器，OpenShift Container Platform 集群将使用更多存储。

集群的限制
可用资源因集群而异。vCenter 中可能的集群数量主要受可用存储空间以及对所需资源数量的限制。确保考虑集群创建的 vCenter 资源的限制和部署集群所需的资源，如 IP 地址和网络。

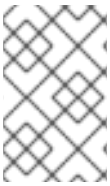
网络要求
网络必须使用 DHCP，并确保 DHCP 服务器被配置为为集群机器提供持久的 IP 地址。



注意

在安装开始前，持久性 IP 地址不可用。分配 DHCP 范围后，在安装后使用持久 IP 地址手动替换分配。

受限网络中的虚拟机必须有权访问 vCenter，以便它可以置备和管理节点、持久性卷声明（PVC）和其他资源。另外，在安装 OpenShift Container Platform 集群前，必须创建以下网络资源：



注意

建议集群中的每个 OpenShift Container Platform 节点都可以访问可通过 DHCP 发现的网络时间协议（NTP）服务器。没有 NTP 服务器也可安装。但是，异步服务器时钟将导致错误，NTP 服务器会阻止。

所需的 IP 地址
安装程序置备的 vSphere 安装需要这些静态 IP 地址：

- API 地址用于访问集群 API。
- Ingress 地址用于集群入口流量。
- 当将集群从版本 4.5 升级到 4.6 时，会使用 control plane 节点地址。

安装 OpenShift Container Platform 集群时，必须向安装程序提供这些 IP 地址。

DNS 记录

您必须在正确的 DNS 服务器中为托管 OpenShift Container Platform 集群的 vCenter 实例创建两个静态 IP 地址的 DNS 记录。在每个记录中，**<cluster_name>** 是集群名称，**<base_domain>** 是您在安装集群时指定的集群基域。完整的 DNS 记录采用如下格式: **<component>.<cluster_name>.<base_domain>.**。

表 12.35. 所需的 DNS 记录

组件	记录	描述
API VIP	**api.<cluster_name>.<base_domain>.**	此 DNS A/AAAA 或 CNAME 记录必须指向 control plane 机器的负载均衡器。此记录必须能由集群外的客户端和集群内的所有节点解析。
Ingress VIP	***.apps.<cluster_name>.<base_domain>.**	通配符 DNS A/AAAA 或 CNAME 记录，指向以运行入口路由器 Pod 的机器（默认为 worker 节点）为目标的负载均衡器。此记录必须能由集群外的客户端和集群内的所有节点解析。

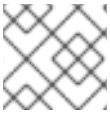
12.4.8. 生成 SSH 私钥并将其添加到代理中

如果要在集群上执行安装调试或灾难恢复，则必须为 **ssh-agent** 和安装程序提供 SSH 密钥。您可以使用此密钥访问公共集群中的 bootstrap 机器来排除安装问题。

注意

在生产环境中，您需要进行灾难恢复和调试。

您可以使用此密钥以 **core** 用户身份通过 SSH 连接到 master 节点。在部署集群时，此密钥会添加到 **core** 用户的 **~/.ssh/authorized_keys** 列表中。



注意

您必须使用一个本地密钥，而不要使用在特定平台上配置的密钥，如 AWS 密钥对。

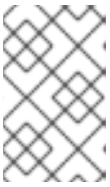
流程

1. 如果还没有为计算机上免密码身份验证而配置的 SSH 密钥，请创建一个。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ ssh-keygen -t ed25519 -N '' \
  -f <path>/<file_name> 1
```

1 指定新 SSH 密钥的路径和文件名，如 **~/.ssh/id_rsa**。如果您已有密钥对，请确保您的公钥位于 **~/.ssh** 目录中。

运行此命令会在指定的位置生成不需要密码的 SSH 密钥。



注意

如果您计划在 **x86_64** 架构中安装使用 FIPS 验证的/Modules in Process 加密库的 OpenShift Container Platform 集群，不要创建使用 **ed25519** 算法的密钥。反之，创建一个使用 **rsa** 或 **ecdsa** 算法的密钥。

2. 作为后台任务启动 **ssh-agent** 进程：

```
$ eval "$(ssh-agent -s)"
```

输出示例

```
Agent pid 31874
```

注意

如果您的集群采用 FIPS 模式，则只使用 FIPS 兼容算法来生成 SSH 密钥。密钥必须是 RSA 或 ECDSA。

3. 将 SSH 私钥添加到 **ssh-agent**：

```
$ ssh-add <path>/<file_name> (1)
```

输出示例

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

(1) 指定 SSH 私钥的路径和文件名，如 **~/.ssh/id_rsa**

后续步骤

- 在安装 OpenShift Container Platform 时，为安装程序提供 SSH 公钥。

12.4.9. 在您的系统信任中添加 vCenter root CA 证书

由于安装程序需要访问 vCenter 的 API，所以必须在安装 OpenShift Container Platform 集群前将 vCenter 的可信 root CA 证书添加到系统信任中。

流程

1. 在 vCenter 主页中下载 vCenter 的 root CA 证书。在 vSphere Web Services SDK 部分点击 **Download trusted root CA certificates**。**<vCenter>/certs/download.zip** 文件下载。

2. 提取包含 vCenter root CA 证书的压缩文件。压缩文件的内容类似以下文件结构：

```
certs
├── lin
│   ├── 108f4d17.0
```

```
│   ├── 108f4d17.r1
│   ├── 7e757f6a.0
│   ├── 8e4f8471.0
│   └── 8e4f8471.r0
├── mac
│   ├── 108f4d17.0
│   ├── 108f4d17.r1
│   ├── 7e757f6a.0
│   ├── 8e4f8471.0
│   └── 8e4f8471.r0
└── win
    ├── 108f4d17.0.crt
    ├── 108f4d17.r1.crl
    ├── 7e757f6a.0.crt
    ├── 8e4f8471.0.crt
    └── 8e4f8471.r0.crl

3 directories, 15 files
```

3. 将您的操作系统的文件添加到系统信任中。例如，在 Fedora 操作系统上运行以下命令：

```
# cp certs/lin/* /etc/pki/ca-trust/source/anchors
```

4. 更新您的系统信任关系。例如，在 Fedora 操作系统上运行以下命令：

```
# update-ca-trust extract
```

12.4.10. 为受限网络安装创建 RHCOS 镜像

下载 Red Hat Enterprise Linux CoreOS（RHCOS）镜像，以便在受限网络 VMware vSphere 环境中安装 OpenShift Container Platform。

先决条件

- 获取 OpenShift Container Platform 安装程序。对于受限网络安装，该程序位于您的镜像 registry 主机上。

流程

1. 登录到红帽客户门户网站的产品下载页。
2. 在 **Version** 下，为 RHEL 8 选择 OpenShift Container Platform 4.6 的最新发行版本。

重要

RHCOS 镜像可能不会随着 OpenShift Container Platform 的每一发行版本都有改变。您必须下载最高版本的镜像，其版本号应小于或等于您安装的 OpenShift Container Platform 版本。如果可用，请使用与 OpenShift Container Platform 版本匹配的镜像版本。

3. 下载 **Red Hat Enterprise Linux CoreOS(RHCOS)- vSphere**镜像。
4. 将您下载的镜像上传到堡垒服务器可访问的位置。

该镜像现在可用于受限安装。记录 OpenShift Container Platform 部署中使用的镜像名称或位置。

12.4.11. 创建安装配置文件

您可以自定义在 VMware vSphere 上安装的 OpenShift Container Platform 集群。

先决条件

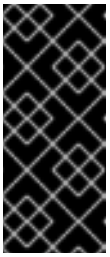
- 获取 OpenShift Container Platform 安装程序以及集群的 pull secret。对于受限网络安装，这些文件位于您的堡垒主机上。
- 具有创建镜像容器镜像仓库（registry）时生成的 **imageContentSources** 值。
- 获取您的镜像 registry 的证书内容。
- 检索 Red Hat Enterprise Linux CoreOS（RHCOS）镜像，并将其上传到可访问的位置。

流程

1. 创建 **install-config.yaml** 文件。
   a. 更改到包含安装程序的目录，再运行以下命令：

```
$ ./openshift-install create install-config --dir <installation_directory> 1
```

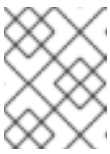
   1 对于 **<installation_directory>**，请指定用于保存安装程序所创建的文件的目录名称。



重要

指定一个空目录。一些安装信息，如 bootstrap X.509 证书，有较短的过期间隔，因此不要重复使用安装目录。如果要重复使用另一个集群安装中的个别文件，可以将其复制到您的目录中。但是，一些安装数据的文件名可能会在发行版本之间有所改变。从 OpenShift Container Platform 老版本中复制安装文件时要格外小心。

   b. 在提示符处，提供您的云的配置详情：
      i. 可选：选择用来访问集群机器的 SSH 密钥。



注意

对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 **ssh-agent** 进程使用的 SSH 密钥。

      ii. 选择 **vsphere** 作为目标平台。
      iii. 指定 vCenter 实例的名称。
      iv. 指定创建集群所需的权限的 vCenter 帐户的用户名和密码。
      安装程序连接到您的 vCenter 实例。
      v. 选择要连接的 vCenter 实例中的数据中心。

vi. 选择要使用的默认 vCenter 数据存储。

vii. 选择要在其中安装 vCenter 集群的 OpenShift Container Platform 集群。安装程序使用 vSphere 集群的 root 资源池作为默认资源池。

viii. 选择包含您配置的虚拟 IP 地址和 DNS 记录的 vCenter 实例中的网络。

ix. 输入您为 control plane API 访问配置的虚拟 IP 地址。

x. 输入您为集群入口配置的虚拟 IP 地址。

xi. 输入基域。这个基域必须与您配置的 DNS 记录中使用的域相同。

xii. 为集群输入一个描述性名称。集群名称必须与您配置的 DNS 记录中使用的相同。

xiii. 粘贴 Red Hat OpenShift Cluster Manager 中的 pull secret 。

2. 在 **install-config.yaml** 文件中，将 **platform.vsphere.clusterOSImage** 的值设置为镜像位置或名称。例如：

```
platform:
  vsphere:
    clusterOSImage: http://mirror.example.com/images/rhcos-43.81.201912131630.0-
vmware.x86_64.ova?
sha256=ffebbd68e8a1f2a245ca19522c16c86f67f9ac8e4e0c1f0a812b068b16f7265d
```

3. 编辑 **install-config.yaml** 文件，以提供在受限网络中安装所需的其他信息。

a. 更新 **pullSecret** 值，使其包含 registry 的身份验证信息：

```
pullSecret: '{"auths":{"<mirror_host_name>:5000": {"auth": "<credentials>","email":
"you@example.com"}}}'
```

对于 **<mirror_host_name>**，请指定您在镜像 registry 证书中指定的 registry 域名；对于 **<credentials>**，请指定您的镜像 registry 的 base64 编码用户名和密码。

b. 添加 **additionalTrustBundle** 参数和值。

```
additionalTrustBundle: |
  -----BEGIN CERTIFICATE-----

ZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZ
  -----END CERTIFICATE-----
```

该值必须是您用于镜像 registry 的证书文件内容，可以是现有的可信证书颁发机构或您为镜像 registry 生成的自签名证书。

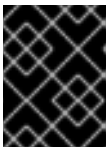
c. 添加镜像内容资源，如下例所示：

```
imageContentSources:
- mirrors:
  - <mirror_host_name>:5000/<repo_name>/release
  source: quay.example.com/openshift-release-dev/ocp-release
- mirrors:
  - <mirror_host_name>:5000/<repo_name>/release
  source: registry.example.com/ocp/release
```

要完成这些值，请使用您在创建镜像容器镜像仓库（registry）时记录的 **imageContentSources**。

4. 对需要的 **install-config.yaml** 文件做任何其他修改。您可以在**安装配置参数**部分中找到有关可用参数的更多信息。

5. 备份 **install-config.yaml** 文件，以便用于安装多个集群。



重要

install-config.yaml 文件会在安装过程中消耗掉。如果要重复使用此文件，必须现在备份。

12.4.11.1. 安装配置参数

在部署 OpenShift Container Platform 集群前，您可以提供参数值，以描述托管集群的云平台的帐户并选择性地自定义集群平台。在创建 **install-config.yaml** 安装配置文件时，您可以通过命令行来提供所需的参数的值。如果要自定义集群，可以修改 **install-config.yaml** 文件来提供关于平台的更多信息。

注意

安装之后，您无法修改 **install-config.yaml** 文件中的这些参数。

重要

openshift-install 命令不验证参数的字段名称。如果指定了不正确的名称，则不会创建相关的文件或对象，且不会报告错误。确保所有指定的参数的字段名称都正确。

12.4.11.1.1. 所需的配置参数

下表描述了所需的安装配置参数：

表 12.36. 所需的参数

参数	描述	值
apiVersion	**install-config.yaml** 内容的 API 版本。当前版本是 **v1**。安装程序还可能支持旧的 API 版本。	字符串
baseDomain	云供应商的基域。此基础域用于创建到 OpenShift Container Platform 集群组件的路由。集群的完整 DNS 名称是 **baseDomain** 和 **metadata.name** 参数值的组合，其格式为 **<metadata.name>.<baseDomain>**。	完全限定域名或子域名，如 **example.com**。

参数	描述	值
metadata	Kubernetes 资源 **ObjectMeta**，其中只消耗 **name** 参数。	对象
metadata.name	集群的名称。集群的 DNS 记录是 **{{.metadata.name}}.{{.baseDomain}}** 的子域。	小写字母和连字符(**-**)的字符串，如 **dev**。
platform	执行安装的具体平台配置：**aws**、**baremetal**、**azure**、**openstack**、**ovirt**、**vsphere**。有关 **platform.<platform>** 参数的额外信息，请参考下表来了解您的具体平台。	对象

pullSecret	从 Red Hat OpenShift Cluster Manager 获取 pull secret，验证从 Quay.io 等服务中下载 OpenShift Container Platform 组件的容器镜像。	<pre>{
		  "auths":{
		  "cloud.openshift.com":{
		  "auth":"b3Blb=",
		  "email":"you@example.com"
		  },
		  "quay.io":{
		  "auth":"b3Blb=",
		  "email":"you@example.com"
		  }
		  }
		}</pre>

12.4.11.1.2. 网络配置参数

您可以根据现有网络基础架构的要求自定义安装配置。例如，您可以扩展集群网络的 IP 地址块，或者提供不同于默认值的不同 IP 地址块。

只支持 IPv4 地址。

表 12.37. 网络参数

参数	描述	值
networking	集群网络的配置。	对象 **注意** 您不能在安装后修改 **networking** 对象指定的参数。

参数	描述	值
networking.networkType	要安装的集群网络供应商 Container Network Interface（CNI）插件。	**OpenShiftSDN** 或 **OVNKubernetes**。默认值为 **OpenShiftSDN**。
networking.clusterNetwork	pod 的 IP 地址块。 默认值为 **10.128.0.0/14**，主机前缀为 **/23**。 如果您指定多个 IP 地址块，则块不得互相重叠。	一个对象数组。例如： networking: clusterNetwork: - cidr: 10.128.0.0/14 hostPrefix: 23
networking.clusterNetwork.cidr	使用 **networking.clusterNetwork** 时需要此项。IP 地址块。 一个 IPv4 网络。	使用 CIDR 形式的 IP 地址块。IPv4 块的前缀长度介于 **0** 到 **32** 之间。
networking.clusterNetwork.hostPrefix	分配给每个单独节点的子网前缀长度。例如，如果 **hostPrefix** 设为 **23**，则每个节点从所给的 **cidr** 中分配一个 **/23** 子网。**hostPrefix** 值 **23** 提供 510（2^(32 - 23)- 2）个 pod IP 地址。	子网前缀。 默认值为 **23**。
networking.serviceNetwork	服务的 IP 地址块。默认值为 **172.30.0.0/16**。 OpenShift SDN 和 OVN-Kubernetes 网络供应商只支持服务网络的一个 IP 地址块。	CIDR 格式具有 IP 地址块的数组。例如： networking: serviceNetwork: - 172.30.0.0/16
networking.machineNetwork	机器的 IP 地址块。 如果您指定多个 IP 地址块，则块不得互相重叠。	一个对象数组。例如： networking: machineNetwork: - cidr: 10.0.0.0/16
networking.machineNetwork.cidr	使用 **networking.machineNetwork** 时需要。IP 地址块。libvirt 以外的所有平台的默认值为 **10.0.0.0/16**。对于 libvirt，默认值为 **192.168.126.0/24**。	CIDR 表示法中的 IP 网络块。 例如：**10.0.0.0/16**。 **注意** 将 **networking.machineNetwork** 设置为与首选 NIC 所在的 CIDR 匹配。

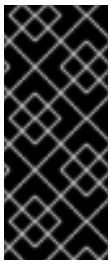
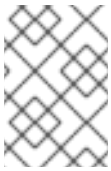
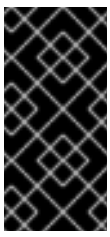
12.4.11.1.3. 可选配置参数

下表描述了可选安装配置参数：

表 12.38. 可选参数

参数	描述	值
additionalTrustBundle	添加到节点可信证书存储中的 PEM 编码 X.509 证书捆绑包。配置了代理时，也可以使用这个信任捆绑包。	字符串
compute	组成计算节点的机器的配置。	machine-pool 对象的数组。详情请查看以下"Machine-pool"表。
compute.architecture	决定池中机器的指令集合架构。目前不支持异构集群，因此所有池都必须指定相同的架构。有效值为 **amd64** （默认值）。	字符串
compute.hyperthreading	是否在计算机器上启用或禁用并发多线程或**超线程**。默认情况下，启用并发多线程以提高机器内核的性能。  **重要** 如果禁用并发多线程，请确保在容量规划时考虑到机器性能可能会显著降低的问题。	**Enabled** 或 **Disabled**
compute.name	使用 **compute** 时需要此值。机器池的名称。	**worker**
compute.platform	使用 **compute** 时需要此值。使用此参数指定托管 worker 机器的云供应商。此参数值必须与 **controlPlane.platform** 参数值匹配。	**aws**、**azure**、**gcp**、**openstack**、**ovirt**、**vsphere** 或 **{}**
compute.replicas	要置备的计算机器数量，也称为 worker 机器。	大于或等于 **2** 的正整数。默认值为 **3**。
controlPlane	组成 control plane 的机器的配置。	**MachinePool** 对象的数组。详情请查看以下"Machine-pool"表。
controlPlane.architecture	决定池中机器的指令集合架构。目前不支持异构集群，因此所有池都必须指定相同的架构。有效值为 **amd64** （默认值）。	字符串

参数	描述	值
controlPlane.hyperthreading	是否在 control plane 机器上启用或禁用并发多线程或**超线程**。默认情况下，启用并发多线程以提高机器内核的性能。  **重要** 如果禁用并发多线程，请确保在容量规划时考虑到机器性能可能会显著降低的问题。	**Enabled** 或 **Disabled**
controlPlane.name	使用 **controlPlane** 时需要。机器池的名称。	**master**
controlPlane.platform	使用 **controlPlane** 时需要。使用此参数指定托管 control plane 机器的云供应商。此参数值必须与 **compute.platform** 参数值匹配。	**aws**、**azure**、**gcp**、**openstack**、**ovirt**、**vsphere** 或 **{}**
controlPlane.replicas	要置备的 control plane 机器数量。	唯一支持的值是 **3**，它是默认值。
credentialsMode	Cloud Credential Operator（CCO）模式。如果没有指定任何模式，CCO 会动态地尝试决定提供的凭证的功能，在支持多个模式的平台上使用 mint 模式。  **注意** 不是所有 CCO 模式都支持所有云供应商。如需有关 CCO 模式的更多信息，请参阅*Red Hat Operator 参考指南*内容中的 *Cloud Credential Operator* 条目。	**Mint**、**Passthrough**、**Manual** 或空字符串(**""**)。

参数	描述	值
fips	启用或禁用 FIPS 模式。默认为 **false**（禁用）。如果启用了 FIPS 模式，运行 OpenShift Container Platform 的 Red Hat Enterprise Linux CoreOS (RHCOS) 机器会绕过默认的 Kubernetes 加密套件，并使用由 RHCOS 提供的加密模块。  **重要** 只有在 **x86_64** 架构中的 OpenShift Container Platform 部署支持 FIPS 验证的/Modules in Process 加密库。  **注意** 如果使用 Azure File 存储，则无法启用 FIPS 模式。	**false** 或 **true**
imageContentSources	release-image 内容的源和仓库。	对象数组。包括一个 **source** 以及可选的 **mirrors**，如下表所示。
imageContentSources.source	使用 **imageContentSources** 时需要。指定用户在镜像拉取规格中引用的仓库。	字符串
imageContentSources.mirrors	指定可能还包含同一镜像的一个或多个仓库。	字符串数组
publish	如何发布或公开集群的面向用户的端点，如 Kubernetes API、OpenShift 路由。	**Internal** 或 **External**。默认值为 **External**。 在非云平台上不支持将此字段设置为 **Internal**。  **重要** 如果将字段的值设为 **Internal**，集群将无法运行。如需更多信息，请参阅 BZ#1953035。

参数	描述	值
sshKey	用于验证集群机器访问的 SSH 密钥或密钥。 **注意** 对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 **ssh-agent** 进程使用的 SSH 密钥。	一个或多个密钥。例如： sshKey: <key1> <key2> <key3>

12.4.11.1.4. 其他 VMware vSphere 配置参数

下表描述了其他 VMware vSphere 配置参数：

表 12.39. 其他 VMware vSphere 集群参数

参数	描述	值
platform.vsphere.vCenter	vCenter 服务器的完全限定主机名或 IP 地址。	字符串
platform.vsphere.username	用于连接 vCenter 实例的用户名。此用户必须至少具有 vSphere 中 静态或动态持久性卷置备 所需的角色和权限。	字符串
platform.vsphere.password	vCenter 用户名的密码。	字符串
platform.vsphere.datacenter	要在 vCenter 实例中使用的数据中心的名称。	字符串
platform.vsphere.defaultDatastore	用于置备卷的默认数据存储名称。	字符串
platform.vsphere.folder	*可选*。安装程序创建虚拟机的现有文件夹的绝对路径。如果没有提供这个值，安装程序会创建一个文件夹，它的名称是数据中心虚拟机文件夹中的基础架构 ID。	字符串，如 **/<datacenter_name>/vm/<folder_name>/<subfolder_name>**。
platform.vsphere.network	包含您配置的虚拟 IP 地址和 DNS 记录的 vCenter 实例中的网络。	字符串
platform.vsphere.cluster	在其中安装 OpenShift Container Platform 集群的 vCenter 集群。	字符串

参数	描述	值
platform.vsphere.apiVIP	为 control plane API 访问配置的虚拟 IP（VIP）地址。	IP 地址，如 **128.0.0.1**。
platform.vsphere.ingressVIP	为集群入口配置的虚拟 IP（VIP）地址。	IP 地址，如 **128.0.0.1**。

12.4.11.1.5. 可选的 VMware vSphere 机器池配置参数

下表描述了可选的 VMware vSphere 机器池配置参数：

表 12.40. 可选的 VMware vSphere 机器池参数

参数	描述	值
platform.vsphere.clusterOSImage	安装程序从中下载 RHCOS 镜像的位置。您必须设置此参数以便在受限网络中执行安装。	HTTP 或 HTTPS URL，可选使用 SHA-256 checksum。例如：**https://mirror.openshift.com/images/rhcos-<version>-vmware.<architecture>.ova**。
platform.vsphere.osDisk.diskSizeGB	以 GB 为单位的磁盘大小。	整数
platform.vsphere.cpus	分配虚拟机的虚拟处理器内核总数。	整数
platform.vsphere.coresPerSocket	虚拟机中每个插槽的内核数。虚拟机上的虚拟套接字数量为 **platform.vsphere.cpus/platform.vsphere.coresPerSocket**。默认值为 **1**。	整数
platform.vsphere.memoryMB	以 MB 为单位的虚拟机内存大小。	整数

12.4.11.2. 安装程序置备的 VMware vSphere 集群的 install-config.yaml 文件示例

您可以自定义 install-config.yaml 文件，以指定有关 OpenShift Container Platform 集群平台的更多信息，或修改所需参数的值。

```
apiVersion: v1
baseDomain: example.com 1
compute: 2
- hyperthreading: Enabled 3
  name: worker
  replicas: 3
  platform:
    vsphere: 4
```

```
      cpus: 2
      coresPerSocket: 2
      memoryMB: 8192
      osDisk:
        diskSizeGB: 120
controlPlane: 5
  hyperthreading: Enabled 6
  name: master
  replicas: 3
  platform:
    vsphere: 7
      cpus: 4
      coresPerSocket: 2
      memoryMB: 16384
      osDisk:
        diskSizeGB: 120
metadata:
  name: cluster 8
platform:
  vsphere:
    vcenter: your.vcenter.server
    username: username
    password: password
    datacenter: datacenter
    defaultDatastore: datastore
    folder: folder
    network: VM_Network
    cluster: vsphere_cluster_name 9
    apiVIP: api_vip
    ingressVIP: ingress_vip
    clusterOSImage: http://mirror.example.com/images/rhcos-48.83.202103221318-0-
vmware.x86_64.ova 10
fips: false
pullSecret: '{"auths":{"<local_registry>": {"auth": "<credentials>","email": "you@example.com"}}}' 11
sshKey: 'ssh-ed25519 AAAA...'
additionalTrustBundle: | 12
  -----BEGIN CERTIFICATE-----
  ZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZ
  -----END CERTIFICATE-----
imageContentSources: 13
- mirrors:
  - <local_registry>/<local_repository_name>/release
  source: quay.io/openshift-release-dev/ocp-release
- mirrors:
  - <local_registry>/<local_repository_name>/release
  source: quay.io/openshift-release-dev/ocp-v4.0-art-dev
```

1 集群的基域。所有 DNS 记录都必须是这个基域的子域，并包含集群名称。

2 5 **controlPlane** 部分是一个单映射，但 **compute** 部分是一系列映射。为满足不同数据结构的要求，**compute** 部分的第一行必须以连字符 **-** 开头，**controlPlane** 部分的第一行则不可以连字符开头。只使用一个 control plane 池。

3 6

是否要启用或禁用并发多线程或超线程。默认情况下，启用并发多线程以提高机器内核的性能。您可以通过将参数值设为 **Disabled** 来禁用。如果您在某些集群机器上禁用并发多线程，则必须在所有集群机器上禁用。

重要

如果禁用并发多线程，请确保在容量规划时考虑到机器性能可能会显著降低的问题。如果您禁用并发多线程，则计算机必须至少使用 8 个 CPU 和 32GB RAM。

4 7 可选：为 compute 和 control plane 机器提供额外的机器池参数配置。

8 您在 DNS 记录中指定的集群名称。

9 要在其中安装 OpenShift Container Platform 集群的 vSphere 集群。安装程序使用 vSphere 集群的 root 资源池作为默认资源池。

10 堡垒服务器可以访问的 Red Hat Enterprise Linux CoreOS（RHCOS）镜像的位置。

11 对于 **<local_registry>**，请指定 registry 域名，以及您的镜像 registry 用来提供内容的可选端口。例如： **registry.example.com** 或者 **registry.example.com:5000**。使用 **<credentials>** 为您生成的镜像 registry 指定 base64 编码的用户名和密码。

12 提供用于镜像 registry 的证书文件内容。

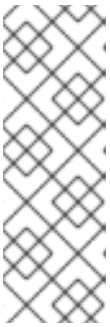
13 提供命令输出中的 **imageContentSources** 部分来镜像存储库。

12.4.11.3. 在安装过程中配置集群范围代理

生产环境可能会拒绝直接访问互联网，而是提供 HTTP 或 HTTPS 代理。您可以通过在 **install-config.yaml** 文件中配置代理设置，将新的 OpenShift Container Platform 集群配置为使用代理。

先决条件

- 您有一个现有的 **install-config.yaml** 文件。
- 您检查了集群需要访问的站点，并决定是否需要绕过代理。默认情况下代理所有集群出口流量，包括对托管云供应商 API 的调用。您需要将站点添加到 **Proxy** 对象的 **spec.noProxy** 字段来绕过代理。



注意

Proxy 对象 **status.noProxy** 字段使用安装配置中的 **networking.machineNetwork[].cidr**、**networking.clusterNetwork[].cidr** 和 **networking.serviceNetwork[]** 字段的值填充。

对于在 Amazon Web Services(AWS)、Google Cloud Platform(GCP)、Microsoft Azure 和 Red Hat OpenStack Platform(RHOSP)上安装, **Proxy** 对象 **status.noProxy** 字段也会使用实例元数据端点填充(**169.254.169.254**)。

流程

1. 编辑 **install-config.yaml** 文件并添加代理设置。例如：

```
apiVersion: v1
```

```
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: example.com 3
additionalTrustBundle: | 4
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
...
```

1. 用于创建集群外 HTTP 连接的代理 URL。URL 必须是 **http**。
2. 用于创建集群外 HTTPS 连接的代理 URL。
3. 要排除在代理中的目标域名、IP 地址或其他网络 CIDR 的逗号分隔列表。在域前面加 **.** 来仅匹配子域。例如： **.y.com** 匹配 **x.y.com**，但不匹配 **y.com**。使用 * 绕过所有目的地的代理。您必须包含 vCenter 的 IP 地址以及用于其机器的 IP 范围。
4. 如果提供，安装程序会在 **openshift-config 命名空间中生成名为 user-ca- bundle** 的配置映射来保存额外的 CA 证书。如果您提供 **additionalTrustBundle** 和至少一个代理设置，则 **Proxy** 对象会被配置为引用 **trustedCA** 字段中的 **user-ca-bundle** 配置映射。然后，Cluster Network Operator 会创建一个 **trusted-ca-bundle** 配置映射，该配置映射将为 **trustedCA** 参数指定的内容与 RHCOS 信任捆绑包合并。 **additionalTrustBundle** 字段是必需的，除非代理的身份证书由来自 RHCOS 信任捆绑包的颁发机构签名。

注意

安装程序不支持代理的 **readinessEndpoints** 字段。

2. 保存该文件，并在安装 OpenShift Container Platform 时引用。

安装程序会创建一个名为 **cluster** 的集群范围代理，该代理使用提供的 **install-config.yaml** 文件中的代理设置。如果没有提供代理设置，仍然会创建一个 **cluster Proxy** 对象，但它会有一个空 **spec**。

注意

只支持名为 **cluster** 的 **Proxy** 对象，且无法创建额外的代理。

12.4.12. 部署集群

您可以在兼容云平台中安装 OpenShift Container Platform。

重要

安装程序的 **create cluster** 命令只能在初始安装过程中运行一次。

先决条件

- 配置托管集群的云平台的帐户。
- 获取 OpenShift Container Platform 安装程序以及集群的 pull secret。

流程

1. 更改为包含安装程序的目录并初始化集群部署：

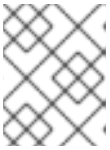
```
$ ./openshift-install create cluster --dir <installation_directory> \ 1
    --log-level=info 2
```

1 对于 **<installation_directory>**，请指定自定义 **./install-config.yaml** 文件的位置。

2 要查看不同的安装详情，请指定 **warn**、**debug** 或 **error**，而不要指定 **info**。

重要

使用托管在 VMC 环境中的堡垒中的 **openshift-install** 命令。



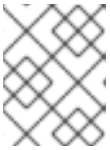
注意

如果您在主机上配置的云供应商帐户没有足够的权限来部署集群，安装过程将会停止，并且显示缺少的权限。

集群部署完成后，终端会显示访问集群的信息，包括指向其 Web 控制台的链接和 **kubeadmin** 用户的凭证。

输出示例

```
...
INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run 'export
KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'
INFO Access the OpenShift web-console here: https://console-openshift-
console.apps.mycluster.example.com
INFO Login to the console with user: "kubeadmin", and password: "4vYBz-Ee6gm-ymBZj-
Wt5AL"
INFO Time elapsed: 36m22s
```

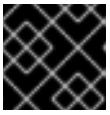


注意

当安装成功时，集群访问和凭证信息还会输出到 **<installation_directory>/.openshift_install.log**。

重要

- 安装程序生成的 Ignition 配置文件包含在 24 小时后过期的证书，然后在过期时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外情况是，您需要手动批准待处理的 **node-bootstrapper** 证书签名请求（CSR）来恢复 kubelet 证书。如需更多信息，请参阅*从过期的 control plane 证书中恢复*的文档。
- 建议您在生成 12 小时后使用 Ignition 配置文件，因为集群安装后 24 小时证书从 16 小时轮转至 22 小时。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中运行证书更新时避免安装失败。



重要

您不得删除安装程序或安装程序所创建的文件。需要这两者才能删除集群。

12.4.13. 通过下载二进制文件安装 OpenShift CLI

您需要安装 CLI（**oc**） 来使用命令行界面与 OpenShift Container Platform 进行交互。您可在 Linux 、Windows 或 macOS 上安装 **oc**。

重要

如果安装了旧版本的 **oc**，则无法使用 OpenShift Container Platform 4.6 中的所有命令。下载并安装新版本的 **oc**。

12.4.13.1. 在 Linux 上安装 OpenShift CLI

您可以按照以下流程在 Linux 上安装 OpenShift CLI（**oc**）二进制文件。

流程

1. 进入到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 在 **Version** 下拉菜单中选择相应的版本。
3. 单击 **OpenShift v4.6 Linux 客户端** 条目旁边的 **Download Now**，再保存文件。
4. 解包存档：

```
$ tar xvzf <file>
```

5. 把 **oc** 二进制代码放到 **PATH** 中的目录中。
   执行以下命令可以查看当前的 **PATH** 设置：

```
$ echo $PATH
```

安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
$ oc <command>
```

12.4.13.2. 在 Windows 上安装 OpenShift CLI

您可以按照以下流程在 Windows 上安装 OpenShift CLI（**oc**）二进制代码。

流程

1. 进入到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 在 **Version** 下拉菜单中选择相应的版本。
3. 单击 **OpenShift v4.6 Windows 客户端**条目旁边的 **Download Now**，再保存文件。
4. 使用 ZIP 程序解压存档。

5. 把 **oc** 二进制代码放到 **PATH** 中的目录中。
   要查看您的 **PATH**，请打开命令提示窗口并执行以下命令：

```
C:\> path
```

安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
C:\> oc <command>
```

12.4.13.3. 在 macOS 上安装 OpenShift CLI

您可以按照以下流程在 macOS 上安装 OpenShift CLI（**oc**）二进制代码。

流程

1. 进入到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 在 **Version** 下拉菜单中选择相应的版本。
3. 单击 **OpenShift v4.6 MacOSX 客户端**条目旁边的 **Download Now**，再保存文件。
4. 解包和解压存档。
5. 将 **oc** 二进制文件移到 PATH 的目录中。
   要查看您的 **PATH**，打开一个终端窗口并执行以下命令：

```
$ echo $PATH
```

安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
$ oc <command>
```

12.4.14. 使用 CLI 登录到集群

您可以通过导出集群 **kubeconfig** 文件，以默认系统用户身份登录集群。**kubeconfig** 文件包含关于集群的信息，供 CLI 用于将客户端连接到正确集群和 API 服务器。该文件特只适用于一个特定的集群，在 OpenShift Container Platform 安装过程中创建。

先决条件

- 已部署了 OpenShift Container Platform 集群。
- 已安装 **oc** CLI。

流程

1. 导出 **kubeadmin** 凭证：

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

2. 使用导出的配置，验证能否成功运行 **oc** 命令：

```
$ oc whoami
```

输出示例

```
system:admin
```

12.4.15. 禁用默认的 OperatorHub 源

在 OpenShift Container Platform 安装过程中，默认为 OperatorHub 配置由红帽和社区项目提供的源内容的 operator 目录。在受限网络环境中，必须以集群管理员身份禁用默认目录。

流程

- 通过在 **OperatorHub** 对象中添加 **disableAllDefaultSources: true** 来禁用默认目录的源：

```
$ oc patch OperatorHub cluster --type json \
    -p '[{"op": "add", "path": "/spec/disableAllDefaultSources", "value": true}]'
```

提示

或者，您可以使用 Web 控制台管理目录源。在 **Administration → Cluster Settings → Global Configuration → OperatorHub** 页面中，点 **Sources** 选项卡，其中可创建、删除、禁用和启用单独的源。

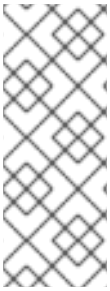
12.4.16. 创建 registry 存储

安装集群后，必须为 Registry Operator 创建存储。

12.4.16.1. 安装过程中删除的镜像 registry

在不提供可共享对象存储的平台上，OpenShift Image Registry Operator bootstraps 本身的状态是 **Removed**。这允许 **openshift-installer** 在这些平台类型上完成安装。

将 **ManagementState** Image Registry Operator 配置从 **Removed** 改为 **Managed**。



注意

Prometheus 控制台提供了一个 **ImageRegistryRemoved** 警报，例如：

"Image Registry has been removed.**ImageStreamTags**, **BuildConfigs** and **DeploymentConfigs** which reference **ImageStreamTags** may not work as expected.Please configure storage and update the config to **Managed** state by editing configs.imageregistry.operator.openshift.io."

12.4.16.2. 镜像 registry 存储配置

对于不提供默认存储的平台，Image Registry Operator 最初将不可用。安装后，您必须配置 registry 使用的存储，这样 Registry Operator 才可用。

示配置生产集群所需的持久性卷的说明。如果适用，显示有关将空目录配置为存储位置的说明，该位置只可用于非生产集群。

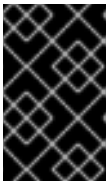
另外还提供了在升级过程中使用 **Recreate** rollout 策略来允许镜像 registry 使用块存储类型的说明。

12.4.16.2.1. 为 VMware vSphere 配置 registry 存储

作为集群管理员，在安装后需要配置 registry 来使用存储。

先决条件

- 具有 Cluster Administrator 权限
- VMware vSphere上有一个集群。
- 为集群置备的持久性存储，如 Red Hat OpenShift Container Storage。



重要

如果您只有一个副本，OpenShift Container Platform 支持对镜像 registry 存储的 **ReadWriteOnce** 访问。要部署支持高可用性的、带有两个或多个副本的镜像 registry，需要 **ReadWriteMany** 访问设置。

- 必须有“ 100Gi”容量。

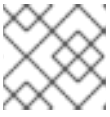
重要

测试显示，在 RHEL 中使用 NFS 服务器作为核心服务的存储后端可能会出现问题。这包括 OpenShift Container Registry 和 Quay，Prometheus 用于监控存储，以及 Elasticsearch 用于日志存储。因此，不推荐使用 RHEL NFS 作为 PV 后端用于核心服务。

市场上的其他 NFS 实现可能没有这些问题。如需了解更多与此问题相关的信息，请联络相关的 NFS 厂商。

流程

1. 为了配置 registry 使用存储，需要修改 **configs.imageregistry/cluster** 资源中的 **spec.storage.pvc** 。



注意

使用共享存储时，请查看您的安全设置以防止被外部访问。

2. 验证您没有 registry pod:

```
$ oc get pod -n openshift-image-registry
```

注意

如果存储类型为 **emptyDIR**，则副本数不能超过 **1**。

3. 检查 registry 配置：

```
$ oc edit configs.imageregistry.operator.openshift.io
```

输出示例

```
storage:
  pvc:
    claim: 1
```

1 将 **claim** 字段留空以允许自动创建一个 **image-registry-storage** PVC。

4. 检查 **clusteroperator** 的状态：

```
$ oc get clusteroperator image-registry
```

12.4.17. OpenShift Container Platform 的 Telemetry 访问

在 OpenShift Container Platform 4.6 中，默认运行的 Telemetry 服务提供有关集群健康状况和成功更新的指标，需要访问互联网。如果您的集群连接到互联网，Telemetry 会自动运行，而且集群会注册到 OpenShift Cluster Manager。

确认 OpenShift Cluster Manager 清单正确后，可以由 Telemetry 自动维护，也可以使用 OpenShift Cluster Manager 手动维护，使用订阅监控来跟踪帐户或多集群级别的 OpenShift Container Platform 订阅。

其他资源

- 有关 Telemetry 服务的更多信息，请参阅关于远程健康监控。

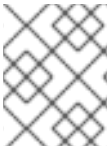
12.4.18. 后续步骤

- 自定义集群。
- 为 Cluster Samples Operator 和 **must-gather** 工具配置镜像流。
- 了解如何在受限网络中使用 Operator Lifecycle Manager（OLM）。
- 如果需要，您可以选择不使用远程健康报告。
- 设置 registry 并配置 registry 存储。

12.5. 使用用户置备的基础架构在 VMC 上安装集群

在 OpenShift Container Platform 版本 4.6 中，您可以在将其部署到 VMware Cloud（VMC）on AWS 来置备的 VMware vSphere 基础架构上安装集群。

为 OpenShift Container Platform 部署配置了 VMC 环境后，您要使用堡垒管理主机中的 OpenShift Container Platform 安装程序，并位于 VMC 环境中。安装程序和 control plane 可以自动部署和管理 OpenShift Container Platform 集群所需的资源。

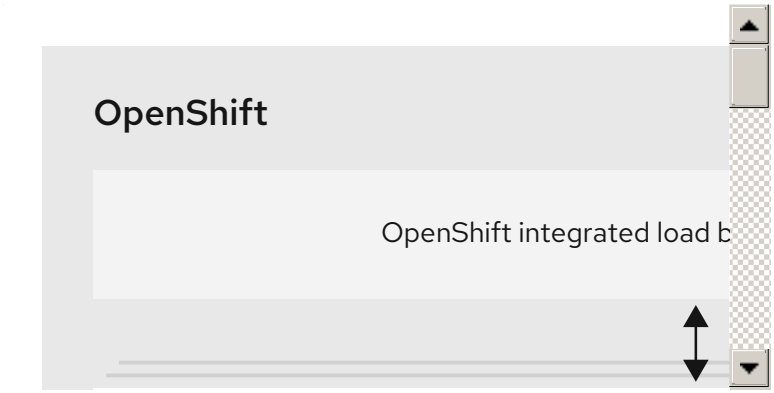


注意

OpenShift Container Platform 支持将集群部署到单个 VMware vCenter 中。不支持在多个 vCenter 上使用机器/机器集部署集群。

12.5.1. 为 vSphere 设置 VMC

您可以在 AWS 托管的 vSphere 集群上安装 OpenShift Container Platform（VMC），以便启用在混合云内部和内部管理应用程序。



在 VMware vSphere 上安装 OpenShift Container Platform 之前，您必须在 VMC 环境中配置多个选项。确定您的 VMC 环境有以下先决条件：

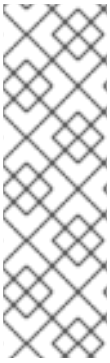
- 创建非专用的、启用了 DHCP 的、NSX-T 网络片段和子网。其他虚拟机（VM）可以托管在子网上，但至少有 8 个 IP 地址可用于 OpenShift Container Platform 部署。
- 配置以下防火墙规则：
  - OpenShift Container Platform 计算网络和互联网间的 ANY:ANY 防火墙规则。节点和应用程序用于下载容器镜像。
  - 在安装主机和软件定义的数据中心（SDDC）管理网络之间的端口 443 的 ANY:ANY 防火墙规则。这可让您在部署过程中上传 Red Hat Enterprise Linux CoreOS（RHCOS）OVA。
  - OpenShift Container Platform 计算网络和 vCenter 间的 HTTPS 防火墙规则。此连接允许 OpenShift Container Platform 与 vCenter 通信，以置备和管理节点、持久性卷声明（PVC）和其他资源。
- 您必须具有以下信息才能部署 OpenShift Container Platform:
  - OpenShift Container Platform 集群名称，如 **vmc-prod-1**。
  - 基本 DNS 名称，如 **companyname.com**。
  - 如果没有使用默认设置，则必须识别 Pod 网络 CIDR 和服务网络 CIDR，默认设置为 **10.128.0.0/14** 和 **172.30.0.0/16**。这些 CIDR 用于 pod 到 pod 和 pod 到服务的通信，且无法在外部访问，但不得与机构中的现有子网重叠。
  - 以下 vCenter 信息：
    - vCenter 主机名、用户名和密码
    - 数据中心名称，如 **SDDC-Datacenter**
    - 集群名称，如 **Cluster-1**

- 网络名称
- 数据存储名称，如 **WorkloadDatastore**

注意

建议在集群安装完成后将 vSphere 集群移到 VMC **Compute-ResourcePool** 资源池。

- 基于 Linux 的主机作为堡垒部署到 VMC。
  - 堡垒主机可以是 Red Hat Enterprise Linux（RHEL）或其他基于 Linux 的主机，它必须具有互联网连接，并可以将 OVA 上传到 ESXi 主机。
  - 将 OpenShift CLI 工具下载并安装到堡垒主机。
    - **openshift-install** 安装程序
    - OpenShift CLI（**oc**）工具



注意

您不能将 VMware NSX Container Plugin 用于 Kubernetes（NCP），NCP 不使用 NSX 作为 OpenShift SDN。目前 VMC 提供的 NSX 版本与 OpenShift Container Platform 认证的 NCP 版本不兼容。

但是，NSX DHCP 服务会通过全堆栈自动 OpenShift Container Platform 部署来管理虚拟机 IP 管理，节点可以手动或自动置备，可通过 Machine API 与 vSphere 集成。另外，还创建 NSX 防火墙规则，以便启用 OpenShift Container Platform 集群以及堡垒主机和 VMC vSphere 主机间的访问。

12.5.1.1. VMC Sizer 工具

AWS 上的 VMware Cloud 基于 AWS 裸机基础架构构建，这与运行 AWS 原生服务的裸机基础架构相同。当在 AWS 软件定义的数据中心（SDDC）上部署 VMware 云时，您要将这些物理服务器节点以单一租户方式运行 VMware ESXi hypervisor。这意味着其他使用 VMC 的用户无法访问物理基础结构。务必要考虑托管您的虚拟基础架构所需的物理主机数量。

要确定这一点，VMware 在 AWS Sizer 上提供 VMC。使用这个工具，您可以定义要在 VMC 上托管的资源：

- 工作负载类型
- 虚拟机总数
- 规格信息，例如：
  - 存储要求
  - vCPUs
  - vRAM
  - 过量使用比例

通过这些信息，sizer 工具可以根据 VMware 最佳实践生成报告，并推荐集群配置和您需要的主机数量。

12.5.2. vSphere 先决条件

- 置备块 registry 存储。如需有关持久性存储的更多信息，请参阅了解持久性存储。
- 查看有关 OpenShift Container Platform 安装和更新流程的详细信息。
- 如果使用防火墙，则必须将其配置为允许集群需要访问的站点。

注意

如果您要配置代理，请务必也要查看此站点列表。

12.5.3. OpenShift Container Platform 的互联网访问

在 OpenShift Container Platform 4.6 中，您需要访问互联网来安装集群。

您必须具有以下互联网访问权限：

- 访问 OpenShift Cluster Manager 以下载安装程序并执行订阅管理。如果集群可以访问互联网，并且没有禁用 Telemetry，该服务会自动授权您的集群。
- 访问 Quay.io，以获取安装集群所需的软件包。
- 获取执行集群更新所需的软件包。

重要

如果您的集群无法直接访问互联网，则可以在置备的某些类基础架构上执行受限网络安装。在此过程中，您要下载所需的内容，并使用它在镜像 registry（mirror registry） 中填充安装集群并生成安装程序所需的软件包。对于某些安装类型，集群要安装到的环境不需要访问互联网。在更新集群之前，要更新 registry 镜像系统中的内容。

12.5.4. VMware vSphere 基础架构要求

您必须在满足您使用的组件要求的 VMware vSphere 版本 6 或 7 实例上安装 OpenShift Container Platform 集群。

表 12.41. VMware 组件支持的最低 vSphere 版本

组件	最低支持版本	描述
虚拟机监控程序	vSphere 6.5 及之后的版本 13	此版本是 Red Hat Enterprise Linux CoreOS(RHCOS)支持的最低版本。请查看 Red Hat Enterprise Linux 8 支持的管理程序列表。
使用 in-tree 驱动程序存储	vSphere 6.5 及之后的版本	此插件使用 OpenShift Container Platform 中包含的 vSphere 的树内存储驱动程序创建 vSphere 存储。

如果您使用 vSphere 版本 6.5 实例，请在安装 OpenShift Container Platform 前考虑升级到 6.7U3 或 7.0。

重要

您必须确保在安装 OpenShift Container Platform 前同步 ESXi 主机上的时间。请参阅 VMware 文档中的编辑主机时间配置。

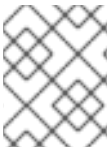
12.5.5. 具有用户置备基础架构的集群的机器要求

对于含有用户置备的基础架构的集群，您必须部署所有所需的机器。

12.5.5.1. 所需的机器

最小的 OpenShift Container Platform 集群需要下列主机：

- 一个临时 bootstrap 机器
- 三台 control plane 或 master 机器
- 至少两台计算机器，也称为 worker 机器。



注意

集群要求 bootstrap 机器在三台 control plane 机器上部署 OpenShift Container Platform 集群。您可在安装集群后删除 bootstrap 机器。

重要

要保持集群的高可用性，请将独立的物理主机用于这些集群机器。

bootstrap 和 control plane 机器必须使用 Red Hat Enterprise Linux CoreOS (RHCOS) 作为操作系统。但是，计算机器可以在 Red Hat Enterprise Linux CoreOS(RHCOS)或 Red Hat Enterprise Linux(RHEL)7.9 之间进行选择。

请注意，RHCOS 基于 Red Hat Enterprise Linux（RHEL） 8，并继承其所有硬件认证和要求。请查看Red Hat Enterprise Linux 技术功能及限制 。

12.5.5.2. 网络连接要求

所有 Red Hat Enterprise Linux CoreOS (RHCOS) 机器在启动过程中需要 **initramfs** 中的网络从 Machine Config Server 获取 Ignition 配置文件。在初次启动过程中，需要一个 DHCP 服务器或设置了静态 IP 地址来建立网络连接，以下载它们的 Ignition 配置文件。另外，集群中的每个 OpenShift Container Platform 节点都必须有权访问网络时间协议（NTP）服务器。如果 DHCP 服务器提供 NTP 服务器信息，Red Hat Enterprise Linux CoreOS（RHCOS）机器上的 chrony 时间服务会读取信息，并可与 NTP 服务器同步时钟。

12.5.5.3. 最低资源要求

每台集群机器都必须满足以下最低要求：

表 12.42. 最低资源要求

机器	操作系统	vCPU [1]	虚拟内存	存储	IOPS [2]
bootstrap	RHCOS	4	16 GB	100 GB	300
Control plane	RHCOS	4	16 GB	100 GB	300
Compute	RHCOS 或 RHEL 7.9	2	8 GB	100 GB	300

1. 当未启用并发多线程 (SMT) 或超线程时，一个 vCPU 相当于一个物理内核。启用后，使用以下公式来计算对应的比例：（每个内核数的线程）× sockets = vCPU。

2. OpenShift Container Platform 和 Kubernetes 对磁盘性能非常敏感，建议使用更快的存储速度，特别是 control plane 节点上需要 10 ms p99 fsync 持续时间的 etcd。请注意，在许多云平台上，存储大小和 IOPS 可一起扩展，因此您可能需要过度分配存储卷来获取足够的性能。

12.5.5.4. 证书签名请求管理

在使用您置备的基础架构时，集群只能有限地访问自动机器管理，因此您必须提供一种在安装后批准集群证书签名请求 (CSR) 的机制。**kube-controller-manager** 只能批准 kubelet 客户端 CSR。 **machine-approver** 无法保证使用 kubelet 凭证请求的提供证书的有效性，因为它不能确认是正确的机器发出了该请求。您必须决定并实施一种方法，以验证 kubelet 提供证书请求的有效性并进行批准。

12.5.6. 创建用户置备的基础架构

在部署采用用户置备的基础架构的 OpenShift Container Platform 集群前，您必须创建底层基础架构。

先决条件

- 在为集群创建支持基础架构之前，请参阅OpenShift Container Platform 4.x Tested Integrations页。

流程

1. 在每个节点上配置 DHCP 或设置静态 IP 地址。
2. 提供所需的负载均衡器。
3. 配置机器的端口。
4. 配置 DNS。
5. 确保网络可以正常工作。

12.5.6.1. 用户置备的基础架构对网络的要求

所有 Red Hat Enterprise Linux CoreOS（RHCOS）机器在启动过程中需要 **initramfs** 中的网络从机器配置服务器获取 Ignition 配置。

在初次启动过程中，需要一个 DHCP 服务器或集群中的每个机器都设置了静态 IP 地址来建立网络连接，以下载它们的 Ignition 配置文件。

建议您使用 DHCP 服务器为集群进行长期机器管理。确保 DHCP 服务器已配置为向集群机器提供持久 IP 地址和主机名。

Kubernetes API 服务器必须能够解析集群机器的节点名称。如果 API 服务器和 worker 节点位于不同的区域中，您可以配置默认 DNS 搜索区域，以便 API 服务器能够解析节点名称。另一种支持的方法是始终在节点对象和所有 DNS 请求中使用完全限定域名来指代主机。

您必须配置机器间的网络连接，以便集群组件进行通信。每台机器都必须能够解析集群中所有其他机器的主机名。

表 12.43. 所有机器到所有机器

协议	端口	描述
ICMP	N/A	网络可访问性测试
TCP	**1936**	指标
	9000-9999	主机级别的服务，包括端口 **9100-9101** 上的节点导出器和端口 **9099** 上的 Cluster Version Operator。
	10250-10259	Kubernetes 保留的默认端口
	10256	openshift-sdn
UDP	**4789**	VXLAN 和 Geneve
	6081	VXLAN 和 Geneve
	9000-9999	主机级别的服务，包括端口 **9100-9101** 上的节点导出器。
TCP/UDP	**30000-32767**	Kubernetes 节点端口

表 12.44. 要通过控制平面的所有机器

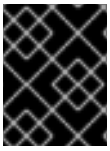
协议	端口	描述
TCP	**6443**	Kubernetes API

表 12.45. control plane 机器到 control plane 机器

协议	端口	描述
TCP	**2379-2380**	etcd 服务器和对等端口

网络拓扑要求

您为集群置备的基础架构必须满足下列网络拓扑要求。



重要

OpenShift Container Platform 要求所有节点都能访问互联网，以便为平台容器提取镜像并向红帽提供遥测数据。

负载均衡器

在安装 OpenShift Container Platform 前，您必须置备两个满足以下要求的负载均衡器：

1. **API 负载均衡器**：提供一个通用端点，供用户（包括人和机器）与平台交互和配置。配置以下条件：

   - 只适用于第 4 层负载均衡。这可被称为 Raw TCP、SSL Passthrough 或者 SSL 桥接模式。如果使用 SSL Bridge 模式，必须为 API 路由启用 Server Name Indication（SNI）。
   - 无状态负载平衡算法。这些选项根据负载均衡器的实现而有所不同。

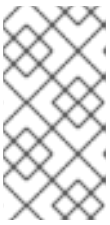
   重要

   不要为 API 负载均衡器配置会话持久性。

   在负载均衡器的前端和后台配置以下端口：

   表 12.46. API 负载均衡器

端口	后端机器（池成员）	内部	外部	描述
6443	Bootstrap 和 control plane.bootstrap 机器初始化集群 control plane 后，您要从负载均衡器中删除 bootstrap 机器。您必须为 API 服务器健康检查探测配置 **/readyz** 端点。	X	X	Kubernetes API 服务器
22623	Bootstrap 和 control plane.bootstrap 机器初始化集群 control plane 后，您要从负载均衡器中删除 bootstrap 机器。	X		机器配置服务器

   注意

   负载均衡器必须配置为，从 API 服务器关闭 **/readyz** 端点到从池中删除 API 服务器实例时最多需要 30 秒。在 **/readyz** 返回错误或处于健康状态后的时间范围内，端点必须被删除或添加。每 5 秒或 10 秒探测一次，有两个成功请求处于健康状态，三个成为不健康的请求经过测试。

2. **应用程序入口负载均衡器**:提供来自集群外部的应用程序流量流量的 Ingress 点。配置以下条件：

   - 只适用于第 4 层负载均衡。这可被称为 Raw TCP、SSL Passthrough 或者 SSL 桥接模式。如果使用 SSL Bridge 模式，您必须为 Ingress 路由启用Server Name Indication（SNI）。
   - 建议根据可用选项以及平台上托管的应用程序类型，使用基于连接的或者基于会话的持久性。

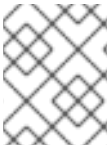
   在负载均衡器的前端和后台配置以下端口：

表 12.47. 应用程序入口负载均衡器

端口	后端机器（池成员）	内部	外部	描述
443	默认运行入口路由器 Pod、计算或 worker 的机器。	X	X	HTTPS 流量
80	默认运行入口路由器 Pod、计算或 worker 的机器。	X	X	HTTP 流量

提示

如果负载均衡器可以看到客户端的真实 IP 地址，启用基于 IP 的会话持久性可提高使用端到端 TLS 加密的应用程序的性能。



注意

OpenShift Container Platform 集群需要正确配置入口路由器。control plane 初始化后，您必须配置入口路由器。

NTP 配置

OpenShift Container Platform 集群默认配置为使用公共网络时间协议（NTP）服务器。如果要使用本地企业 NTP 服务器，或者集群部署在断开连接的网络中，您可以将集群配置为使用特定的时间服务器。如需更多信息，请参阅*配置 chrony 时间服务*的文档。

如果 DHCP 服务器提供 NTP 服务器信息，Red Hat Enterprise Linux CoreOS（RHCOS）机器上的 chrony 时间服务会读取信息，并可与 NTP 服务器同步时钟。

12.5.6.2. 用户置备 DNS 要求

DNS 用于名称解析和反向名称解析。DNS A/AAAA 或 CNAME 记录用于名称解析，PTR 记录用于反向解析名称。反向记录很重要，因为 Red Hat Enterprise Linux CoreOS（RHCOS）使用反向记录为所有节点设置主机名。另外，反向记录用于生成 OpenShift Container Platform 需要操作的证书签名请求（CSR）。

采用用户置备的基础架构的 OpenShift Container Platform 集群需要以下 DNS 记录。在每一记录中，**<cluster_name>** 是集群名称，**<base_domain>** 则是您在 **install-config.yaml** 文件中指定的集群基域。完整的 DNS 记录采用如下格式: **<component>.<cluster_name>.<base_domain>.**。

表 12.48. 所需的 DNS 记录

组件	记录	描述
Kubernetes API	**api.<cluster_name>.<base_domain>.**	添加 DNS A/AAAA 或 CNAME 记录，以及 DNS PTR 记录，以识别 control plane 机器的负载均衡器。这些记录必须由集群外的客户端以及集群中的所有节点解析。

组件	记录	描述
	api-int.<cluster_name>. <base_domain>.	添加 DNS A/AAAA 或 CNAME 记录，以及 DNS PTR 记录，以识别 control plane 机器的负载均衡器。这些记录必须可以从集群中的所有节点解析。 **重要** API 服务器必须能够根据在 Kubernetes 中记录的主机名解析 worker 节点。如果 API 服务器无法解析节点名称，则代理的 API 调用会失败，且您无法从 pod 检索日志。
Routes	***.apps.<cluster_name>. <base_domain>.**	添加通配符 DNS A/AAAA 或 CNAME 记录，指向以运行入口路由器 Pod 的机器（默认为 worker 节点）为目标的负载均衡器。这些记录必须由集群外的客户端以及集群中的所有节点解析。
bootstrap	**bootstrap.<cluster_name>. <base_domain>.**	添加 DNS A/AAAA 或 CNAME 记录，以及 DNS PTR 记录来识别 bootstrap 机器。这些记录必须由集群中的节点解析。
Master 主机	**<master><n>. <cluster_name>. <base_domain>.**	DNS A/AAAA 或 CNAME 记录，以识别 control plane 节点（也称为 master 节点）的每台机器。这些记录必须由集群中的节点解析。
Worker 主机	**<worker><n>. <cluster_name>. <base_domain>.**	添加 DNS A/AAAA 或 CNAME 记录，以识别 worker 节点的每台机器。这些记录必须由集群中的节点解析。

提示

您可以使用 **nslookup <hostname>** 命令来验证名称解析。您可以使用 **dig -x <ip_address>** 命令来验证 PTR 记录的反向名称解析。

下面的 BIND 区文件的例子展示了关于名字解析的 A 记录的例子。这个示例的目的是显示所需的记录。这个示例不是为选择一个名称解析服务提供建议。

例 12.9. DNS 区数据库示例

```
$TTL 1W
@ IN SOA ns1.example.com. root (
   2019070700 ; serial
   3H  ; refresh (3 hours)
   30M  ; retry (30 minutes)
   2W  ; expiry (2 weeks)
   1W )  ; minimum (1 week)
 IN NS ns1.example.com.
 IN MX 10 smtp.example.com.
;
```

```
;
ns1 IN A 192.168.1.5
smtp IN A 192.168.1.5
;
helper IN A 192.168.1.5
helper.ocp4 IN A 192.168.1.5
;
; The api identifies the IP of your load balancer.
api.ocp4  IN A 192.168.1.5
api-int.ocp4  IN A 192.168.1.5
;
; The wildcard also identifies the load balancer.
*.apps.ocp4  IN A 192.168.1.5
;
; Create an entry for the bootstrap host.
bootstrap.ocp4 IN A 192.168.1.96
;
; Create entries for the master hosts.
master0.ocp4  IN A 192.168.1.97
master1.ocp4  IN A 192.168.1.98
master2.ocp4  IN A 192.168.1.99
;
; Create entries for the worker hosts.
worker0.ocp4  IN A 192.168.1.11
worker1.ocp4  IN A 192.168.1.7
;
;EOF
```

下面的 BIND 区文件示例显示了反向名字解析的 PTR 记录示例。

例 12.10. 反向记录的 DNS 区数据库示例

```
$TTL 1W
@ IN SOA ns1.example.com. root (
   2019070700 ; serial
   3H  ; refresh (3 hours)
   30M  ; retry (30 minutes)
   2W  ; expiry (2 weeks)
   1W )  ; minimum (1 week)
 IN NS ns1.example.com.
;
; The syntax is "last octet" and the host must have an FQDN
; with a trailing dot.
97 IN PTR master0.ocp4.example.com.
98 IN PTR master1.ocp4.example.com.
99 IN PTR master2.ocp4.example.com.
;
96 IN PTR bootstrap.ocp4.example.com.
;
5 IN PTR api.ocp4.example.com.
5 IN PTR api-int.ocp4.example.com.
;
11 IN PTR worker0.ocp4.example.com.
```

```
7 IN PTR worker1.ocp4.example.com.
;
;EOF
```

12.5.7. 生成 SSH 私钥并将其添加到代理中

如果要在集群上执行安装调试或灾难恢复，则必须为 **ssh-agent** 和安装程序提供 SSH 密钥。您可以使用此密钥访问公共集群中的 bootstrap 机器来排除安装问题。

注意

在生产环境中，您需要进行灾难恢复和调试。

您可以使用此密钥以 **core** 用户身份通过 SSH 连接到 master 节点。在部署集群时，此密钥会添加到 **core** 用户的 **~/.ssh/authorized_keys** 列表中。

注意

您必须使用一个本地密钥，而不要使用在特定平台上配置的密钥，如 AWS 密钥对。

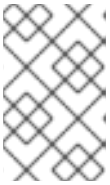
流程

1. 如果还没有为计算机上免密码身份验证而配置的 SSH 密钥，请创建一个。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ ssh-keygen -t ed25519 -N '' \
  -f <path>/<file_name> 1
```

1 指定新 SSH 密钥的路径和文件名，如 **~/.ssh/id_rsa**。如果您已有密钥对，请确保您的公钥位于 **~/.ssh** 目录中。

运行此命令会在指定的位置生成不需要密码的 SSH 密钥。



注意

如果您计划在 **x86_64** 架构中安装使用 FIPS 验证的/Modules in Process 加密库的 OpenShift Container Platform 集群，不要创建使用 **ed25519** 算法的密钥。反之，创建一个使用 **rsa** 或 **ecdsa** 算法的密钥。

2. 作为后台任务启动 **ssh-agent** 进程：

```
$ eval "$(ssh-agent -s)"
```

输出示例

```
Agent pid 31874
```

注意

如果您的集群采用 FIPS 模式，则只使用 FIPS 兼容算法来生成 SSH 密钥。密钥必须是 RSA 或 ECDSA。

3. 将 SSH 私钥添加到 **ssh-agent**：

```
$ ssh-add <path>/<file_name> 1
```

输出示例

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

1 指定 SSH 私钥的路径和文件名，如 **~/.ssh/id_rsa**

后续步骤

- 在安装 OpenShift Container Platform 时，为安装程序提供 SSH 公钥。如果在您置备的基础架构上安装集群，您必须将此密钥提供给集群的机器。

12.5.8. 获取安装程序

在安装 OpenShift Container Platform 之前，将安装文件下载到本地计算机上。

先决条件

- 运行 Linux 或 macOS 的计算机，本地磁盘空间为 500 MB

流程

1. 访问 OpenShift Cluster Manager 站点的 Infrastructure Provider 页面。如果您有红帽帐号，请使用自己的凭证登录。如果没有，请创建一个帐户。
2. 选择您的基础架构供应商。
3. 进入适用于您的安装类型的页面，下载您的操作系统的安装程序，并将文件放在要保存安装配置文件的目录中。。

重要

安装程序会在用来安装集群的计算机上创建若干文件。在完成集群安装后，您必须保留安装程序和安装程序所创建的文件。这两个文件都需要删除集群。

重要

删除安装程序创建的文件不会删除您的集群，即使集群在安装过程中失败也是如此。要删除集群，为特定云供应商完成 OpenShift Container Platform 卸载流程。

4. 提取安装程序。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ tar xvf openshift-install-linux.tar.gz
```

5. 从 Red Hat OpenShift Cluster Manager 下载安装 pull secret 。通过此 pull secret，您可以进行所含授权机构提供的服务的身份验证，这些服务包括为 OpenShift Container Platform 组件提供容器镜像的 Quay.io。

12.5.9. 手动创建安装配置文件

对于使用用户置备的基础架构的 OpenShift Container Platform 安装，您必须手动生成安装配置文件。

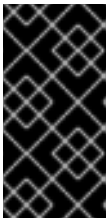
先决条件

- 获取 OpenShift Container Platform 安装程序和集群的访问令牌。

流程

1. 创建用来存储您所需的安装资产的安装目录：

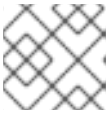
```
$ mkdir <installation_directory>
```



重要

您必须创建目录。一些安装信息，如 bootstrap X.509 证书，有较短的过期间隔，因此不要重复使用安装目录。如果要重复使用另一个集群安装中的个别文件，可以将其复制到您的目录中。但是，一些安装数据的文件名可能会在发行版本之间有所改变。从 OpenShift Container Platform 老版本中复制安装文件时要格外小心。

2. 自定义以下 **install-config.yaml** 文件模板，并将它保存到 **<installation_directory>** 中。



注意

此配置文件必须命名为 **install-config.yaml**。

3. 备份 **install-config.yaml** 文件，以便用于安装多个集群。

重要

install-config.yaml 文件会在安装过程的下一步骤中消耗掉。现在必须备份它。

12.5.9.1. VMware vSphere install-config.yaml 文件示例

您可以自定义 **install-config.yaml** 文件，以指定有关 OpenShift Container Platform 集群平台的更多信息，或修改所需参数的值。

```
apiVersion: v1
baseDomain: example.com (1)
compute:
- hyperthreading: Enabled (2) (3)
  name: worker
  replicas: 0 (4)
controlPlane:
  hyperthreading: Enabled (5) (6)
  name: master
  replicas: 3 (7)
```

```
metadata:
  name: test 8
platform:
  vsphere:
    vcenter: your.vcenter.server 9
    username: username 10
    password: password 11
    datacenter: datacenter 12
    defaultDatastore: datastore 13
    folder: "/<datacenter_name>/vm/<folder_name>/<subfolder_name>" 14
fips: false 15
pullSecret: '{"auths": ...}' 16
sshKey: 'ssh-ed25519 AAAA...' 17
```

1 集群的基域。所有 DNS 记录都必须是这个基域的子域，并包含集群名称。

2 5 **controlPlane** 部分是一个单映射，但 compute 部分是一系列映射。为满足不同数据结构的要求，**compute** 部分的第一行必须以连字符 **-** 开头，**controlPlane** 部分的第一行则不可以连字符开头。虽然这两个部分目前都定义单个机器池，但未来的 OpenShift Container Platform 版本可能会支持在安装过程中定义多个计算池。只使用一个 control plane 池。

3 6 是否要启用或禁用并发多线程或**超线程**。默认情况下，启用并发多线程以提高机器内核的性能。您可以通过将参数值设为 **Disabled** 来禁用。如果您在某些集群机器上禁用并发多线程，则必须在所有集群机器上禁用。

重要

如果禁用并发多线程，请确保在容量规划时考虑到机器性能可能会显著降低的问题。如果您禁用并发多线程，则计算机必须至少使用 8 个 CPU 和 32GB RAM。

4 **replicas** 参数的值必须设置为 **0**。此参数控制集群为您创建和管理的 worker 数量，使用用户置备的基础架构时集群不会执行这些功能。在完成 OpenShift Container Platform 安装前，您必须手动为集群部署 worker 机器。

7 您添加到集群的 control plane 机器数量。由于集群将这个值用作集群中 etcd 端点的数量，因此该值必须与您部署的 control plane 机器数量匹配。

8 您在 DNS 记录中指定的集群名称。

9 vCenter 服务器的完全限定主机名或 IP 地址。

10 用于访问服务器的用户名。此用户必须至少具有 vSphere 中 静态或动态持久性卷置备 所需的角色和权限。

11 与 vSphere 用户关联的密码。

12 vSphere 数据中心。

13 要使用的默认 vSphere 数据存储。

14 可选：对于安装程序置备的基础架构，安装程序创建虚拟机的现有文件夹的绝对路径，如 **/<datacenter_name>/vm/<folder_name>/<subfolder_name>**。如果没有提供这个值，安装程序会在数据中心虚拟机文件夹中创建一个顶层文件夹，其名称为基础架构 ID。如果您为集群提供基础架构,请省略此参数。

15 是否启用或禁用 FIPS 模式。默认情况下不启用 FIPS 模式。如果启用了 FIPS 模式，运行 OpenShift Container Platform 的 Red Hat Enterprise Linux CoreOS (RHCOS) 机器会绕过默认的 Kubernetes

重要

只有在 **x86_64** 架构中的 OpenShift Container Platform 部署支持 FIPS 验证的/Modules in Process 加密库。

16 从 OpenShift Cluster Manager 获取的 pull secret。通过此 pull secret，您可以进行所含授权机构提供的服务的身份验证，这些服务包括为 OpenShift Container Platform 组件提供容器镜像的 Quay.io。

17 Red Hat Enterprise Linux CoreOS (RHCOS) 中 **core** 用户的默认 SSH 密钥的公钥部分。

12.5.9.2. 在安装过程中配置集群范围代理

生产环境可能会拒绝直接访问互联网，而是提供 HTTP 或 HTTPS 代理。您可以通过在 **install-config.yaml** 文件中配置代理设置，将新的 OpenShift Container Platform 集群配置为使用代理。

先决条件

- 您有一个现有的 **install-config.yaml** 文件。
- 您检查了集群需要访问的站点，并决定是否需要绕过代理。默认情况下代理所有集群出口流量，包括对托管云供应商 API 的调用。您需要将站点添加到 **Proxy** 对象的 **spec.noProxy** 字段来绕过代理。

注意

Proxy 对象 **status.noProxy** 字段使用安装配置中的 **networking.machineNetwork[].cidr**、**networking.clusterNetwork[].cidr** 和 **networking.serviceNetwork[]** 字段的值填充。

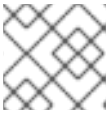
对于在 Amazon Web Services(AWS)、Google Cloud Platform(GCP)、Microsoft Azure 和 Red Hat OpenStack Platform(RHOSP)上安装，**Proxy** 对象 **status.noProxy** 字段也会使用实例元数据端点填充(**169.254.169.254**)。

流程

1. 编辑 **install-config.yaml** 文件并添加代理设置。例如：

```
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: example.com 3
additionalTrustBundle: | 4
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
...
```

1. 用于创建集群外 HTTP 连接的代理 URL。URL 必须是 **http**。
2. 用于创建集群外 HTTPS 连接的代理 URL。
3. 要排除在代理中的目标域名、IP 地址或其他网络 CIDR 的逗号分隔列表。在域前面加 **.** 来仅匹配子域。例如： **.y.com** 匹配 **x.y.com**，但不匹配 **y.com**。使用 ***** 绕过所有目的地的代理。您必须包含 vCenter 的 IP 地址以及用于其机器的 IP 范围。
4. 如果提供，安装程序会在 **openshift-config 命名空间中生成名为 user-ca- bundle** 的配置映射来保存额外的 CA 证书。如果您提供 **additionalTrustBundle** 和至少一个代理设置，则 **Proxy** 对象会被配置为引用 **trustedCA** 字段中的 **user-ca-bundle** 配置映射。然后，Cluster Network Operator 会创建一个 **trusted-ca-bundle** 配置映射，该配置映射将为 **trustedCA** 参数指定的内容与 RHCOS 信任捆绑包合并。**additionalTrustBundle** 字段是必需的，除非代理的身份证书由来自 RHCOS 信任捆绑包的颁发机构签名。



注意

安装程序不支持代理的 **readinessEndpoints** 字段。

2. 保存该文件，并在安装 OpenShift Container Platform 时引用。

安装程序会创建一个名为 **cluster** 的集群范围代理，该代理使用提供的 **install-config.yaml** 文件中的代理设置。如果没有提供代理设置，仍然会创建一个 **cluster Proxy** 对象，但它会有一个空 **spec**。

注意

只支持名为 **cluster** 的 **Proxy** 对象，且无法创建额外的代理。

12.5.10. 创建 Kubernetes 清单和 Ignition 配置文件

由于您必须修改一些集群定义文件并要手动启动集群机器，因此您必须生成 Kubernetes 清单和 Ignition 配置文件，集群需要这两项来创建其机器。

安装配置文件转换为 Kubernetes 清单。清单嵌套到 Ignition 配置文件中，稍后用于创建集群。

重要

- 安装程序生成的 Ignition 配置文件包含在 24 小时后过期的证书，然后在过期时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外情况是，您需要手动批准待处理的 **node-bootstrapper** 证书签名请求（CSR）来恢复 kubelet 证书。如需更多信息，请参阅 *从过期的 control plane 证书中恢复*的文档。
- 建议您在生成 12 小时后使用 Ignition 配置文件，因为集群安装后 24 小时证书从 16 小时轮转至 22 小时。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中运行证书更新时避免安装失败。

先决条件

- 已获得 OpenShift Container Platform 安装程序。
- 已创建 **install-config.yaml** 安装配置文件。

流程

流程

1. 切换到包含安装程序的目录，并为集群生成 Kubernetes 清单：

```
$ ./openshift-install create manifests --dir <installation_directory> 1
```

1 对于 **<installation_directory>**，请指定含有您创建的 **install-config.yaml** 文件的安装目录。

2. 删除定义 control plane 机器的 Kubernetes 清单文件以及计算机器集：

```
$ rm -f openshift/99_openshift-cluster-api_master-machines-*.yaml openshift/99_openshift-cluster-api_worker-machineset-*.yaml
```

由于您要自行创建和管理这些资源，因此不必初始化这些资源。

- 您可以使用机器 API 来保留机器集文件来创建计算机器，但您必须更新对其的引用，以匹配您的环境。

3. 检查 **<installation_directory>/manifests/cluster-scheduler-02-config.yml** Kubernetes 清单文件中的 **mastersSchedulable** 参数是否已设置为 **false**。此设置可防止在 control plane 机器上调度 pod:

   a. 打开 **<installation_directory>/manifests/cluster-scheduler-02-config.yml** 文件。

   b. 找到 **mastersSchedulable** 参数并确保它被设置为 **false**。

   c. 保存并退出文件。

4. 要创建 Ignition 配置文件，从包含安装程序的目录运行以下命令：

```
$ ./openshift-install create ignition-configs --dir <installation_directory> 1
```

1 对于 **<installation_directory>**，请指定相同的安装目录。

该目录中将生成以下文件：

```
.
├── auth
│   ├── kubeadmin-password
│   └── kubeconfig
├── bootstrap.ign
├── master.ign
├── metadata.json
└── worker.ign
```

12.5.11. 提取基础架构名称

Ignition 配置文件包含一个唯一的集群标识符,您可以使用它在 VMware Cloud 中唯一地标识您的集群。如果计划使用集群标识符作为虚拟机文件夹的名称,您必须提取它。

先决条件

- 获取 OpenShift Container Platform 安装程序以及集群的 pull secret。

- 已为集群生成 Ignition 配置文件。
- 安装了 **jq** 软件包。

流程

- 要从 Ignition 配置文件元数据中提取和查看基础架构名称，请运行以下命令：

```
$ jq -r .infraID <installation_directory>/metadata.json 1
```

1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

输出示例

```
openshift-vw9j6 1
```

1 此命令的输出是您的集群名称和随机字符串。

12.5.12. 在 vSphere 中创建 Red Hat Enterprise Linux CoreOS (RHCOS) 机器

在 VMware vSphere 上安装包含用户置备基础架构的集群前，您必须在 vSphere 主机上创建 RHCOS 机器供其使用。

先决条件

- 已获取集群的 Ignition 配置文件。
- 您可以从计算机访问 HTTP 服务器，并且您创建的机器也可以访问该服务器。
- 您已创建了 vSphere 集群。

流程

1. 将名为 **<installation_directory>/bootstrap.ign** 的 bootstrap Ignition 配置文件上传到 HTTP 服务器，该配置文件是由安装程序创建的。记下此文件的 URL。
2. 将 bootstrap 节点的以下辅助 Ignition 配置文件保存到计算机中，存为 **<installation_directory>/merge-bootstrap.ign** ：

```
{
  "ignition": {
    "config": {
      "merge": [
        {
          "source": "<bootstrap_ignition_config_url>", 1
          "verification": {}
        }
      ]
    },
    "timeouts": {},
    "version": "3.1.0"
  },
```

```
  "networkd": {},
  "passwd": {},
  "storage": {},
  "systemd": {}
}
```

1 指定您托管的 bootstrap Ignition 配置文件的 URL。

为 bootstrap 机器创建虚拟机 (VM) 时，您要使用此 Ignition 配置文件。

3. 找到安装程序创建的以下 Ignition 配置文件：
   - **<installation_directory>/master.ign**
   - **<installation_directory>/worker.ign**
   - **<installation_directory>/merge-bootstrap.ign**
4. 将 Ignition 配置文件转换为 Base64 编码。在此流程中，您必须将这些文件添加到虚拟机中的额外配置参数 **guestinfo.ignition.config.data** 中。
   例如，如果您使用 Linux 操作系统，可以使用 **base64** 命令来编码这些文件。

```
$ base64 -w0 <installation_directory>/master.ign > <installation_directory>/master.64
```

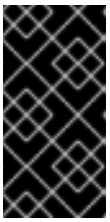
```
$ base64 -w0 <installation_directory>/worker.ign > <installation_directory>/worker.64
```

```
$ base64 -w0 <installation_directory>/merge-bootstrap.ign > <installation_directory>/merge-bootstrap.64
```

重要

如果您计划在安装完成后在集群中添加更多计算机器，请不要删除这些文件。

5. 获取 RHCOS OVA 镜像。镜像位于 RHCOS 镜像镜像页面。



重要

RHCOS 镜像可能不会随着 OpenShift Container Platform 的每一发行版本都有改变。您必须下载一个最高版本的镜像，其版本号应小于或等于您安装的 OpenShift Container Platform 版本。如果可用，请使用与 OpenShift Container Platform 版本匹配的镜像版本。

文件名包含 OpenShift Container Platform 版本号，格式为 **rhcos-vmware.<architecture>.ova**。

6. 在 vSphere 客户端中，在数据中心中创建一个文件夹来存储您的虚拟机。
   a. 点击 **VMs and Templates** 视图。
   b. 右键点击您的数据中心名称。
   c. 点击 **New Folder → New VM and Template Folder**。

d. 在显示的窗口中输入文件夹名称。如果您没有在 **install-config.yaml** 文件中指定现有文件夹，请创建一个文件夹，其名称与基础架构 ID 相同。您可以使用这个文件夹名称，因此 vCenter 会在适当的位置为 Workspace 配置动态置备存储。

7. 在 vSphere 客户端中，为 OVA 镜像创建一个模板，然后根据需要克隆模板。

注意

在以下步骤中，您将创建一个模板，然后克隆所有集群机器的模板。然后，在置备虚拟机时，为该克隆的机器类型提供 Ignition 配置文件的位置。

a. 在 **Hosts and Clusters** 选项卡中，右键点击您的集群名称并选择 **Deploy OVF Template**。

b. 在 **Select an OVF** 选项卡中，指定您下载的 RHCOS OVA 文件的名称。

c. 在 **Select a name and folder** 选项卡中，为您的模板设置 **虚拟机名称**，如 **Template-RHCOS**。点击 vSphere 集群的名称并选择您在上一步中创建的文件夹。

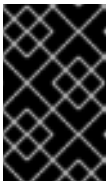
d. 在 **Select a compute resource** 选项卡中，点击您的 vSphere 集群名称。

e. 在 **Select storage** 选项卡中，配置虚拟机的存储选项。

- 根据您的存储要求，选择 **Thin Provision** 或 **Thick Provision**。
- 选择您在 **install-config.yaml** 文件中指定的数据存储。

f. 在 **Select network** 选项卡中，指定您为集群配置的网络（如果可用）。

g. 在创建 OVF 模板时，请不要在 **Customize template** 选项卡上指定值，或者不要再配置模板。



重要

不要启动原始虚拟机模板。VM 模板必须保持关闭状态，必须为新的 RHCOS 机器克隆。启动虚拟机模板会将虚拟机模板配置为平台上的虚拟机，这样可防止它被用作计算机集可以应用配置的模板。

8. 部署模板后，为集群中的机器部署虚拟机。

a. 右键点击模板的名称，再点击 **Clone → Clone to Virtual Machine**。

b. 在 **Select a name and folder** 选项卡中，指定虚拟机的名称。名称中可以包括机器类型，如 **control-plane-0** 或 **compute-1**。

c. 在 **Select a name and folder** 选项卡中，选择您为集群创建的文件夹名称。

d. 在 **Select a compute resource** 选项卡中，选择数据中心中的主机名称。
对于 bootstrap 机器，指定您托管的 bootstrap Ignition 配置文件的 URL。

e. 可选：在 **Select storage** 选项卡中，自定义存储选项。

f. 在 **Select clone options** 中，选择 **Customize this virtual machine's hardware**。

g. 在 **Customize hardware** 选项卡中，点击 **VM Options → Advanced**。

- 可选：覆盖 vSphere 中的默认 DHCP 网络。启用静态 IP 网络：

i. 设置静态 IP 配置：

```
$ export IPCFG="ip=<ip>::<gateway>:<netmask>:<hostname>:<iface>:none
nameserver=srv1 [nameserver=srv2 [nameserver=srv3 [...]]]"
```

示例命令

```
$ export IPCFG="ip=192.168.100.101::192.168.100.254:255.255.255.0:::none
nameserver=8.8.8.8"
```

ii. 在从 vSphere 中的 OVA 引导虚拟机前，设置 **guestinfo.afterburn.initrd.network-kargs** 属性：

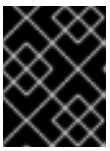
```
$ govc vm.change -vm "<vm_name>" -e "guestinfo.afterburn.initrd.network-
kargs=${IPCFG}"
```

- 可选：在出现集群性能问题时，从 **Latency Sensitivity** 列表中选择 **High**。确定虚拟机的 CPU 和内存保留有以下值：
  - 内存保留值必须等于其配置的内存大小。
  - CPU 保留值必须至少是低延迟虚拟 CPU 的数量，乘以测量的物理 CPU 速度。
- 点击 **Edit Configuration**，然后在 **Configuration Parameters** 窗口中点击 **Add Configuration Params**。定义以下参数名称和值：
  - **guestinfo.ignition.config.data** ：找到您在此流程中创建的 base-64 编码文件，并粘贴此机器类型的 base64 编码 Ignition 配置文件的内容。
  - **guestinfo.ignition.config.data.encoding**：指定 **base64**。
  - **disk.EnableUUID**：指定 **TRUE**。

h. 在 **Customize hardware** 选项卡的 **Virtual Hardware** 面板中，根据需要修改指定的值。确保 RAM、CPU 和磁盘存储的数量满足机器类型的最低要求。

i. 完成配置并打开虚拟机电源。

9. 对于每台机器，按照前面的步骤为集群创建其余的机器。



重要

此刻您必须创建 bootstrap 和 control plane 机器。由于计算机器中已默认部署了一些 Pod，因此在安装集群前，还要创建至少两台计算机器。

12.5.13. 在 vSphere 中创建更多 Red Hat Enterprise Linux CoreOS（RHCOS）机器

您可以为集群创建更多计算机器，在 VMware vSphere 上使用用户置备的基础架构。

先决条件

- 获取计算机器的 Base64 编码 Ignition 文件。
- 您可以访问您为集群创建的 vSphere 模板。

流程

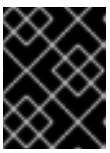
1. 部署模板后，为集群中的机器部署虚拟机。
   a. 右键点击模板的名称，再点击 **Clone → Clone to Virtual Machine**。
   b. 在 **Select a name and folder**选项卡中，指定虚拟机的名称。您可以在名称中包含机器类型，如 **compute-1**。
   c. 在 **Select a name and folder**选项卡中，选择您为集群创建的文件夹名称。
   d. 在 **Select a compute resource** 选项卡中，选择数据中心中的主机名称。
   e. 可选：在 **Select storage** 选项卡中，自定义存储选项。
   f. 在 **Select clone options** 中，选择 **Customize this virtual machine's hardware**。
   g. 在 **Customize hardware** 选项卡中，点击 **VM Options → Advanced**。
      - 从 **Latency Sensitivity** 列表中选择 **High**。
      - 点击 **Edit Configuration**，然后在 **Configuration Parameters** 窗口中点击 **Add Configuration Params**。定义以下参数名称和值：
        - **guestinfo.ignition.config.data**：粘贴此机器类型的 Base64 编码计算 Ignition 配置文件的内容。
        - **guestinfo.ignition.config.data.encoding**：指定 **base64**。
        - **disk.EnableUUID**：指定 **TRUE**。
   h. 在 **Customize hardware** 选项卡的 **Virtual Hardware** 面板中，根据需要修改指定的值。确保 RAM、CPU 和磁盘存储的数量满足机器类型的最低要求。另外，如果有多个可用的网络，请确定在 **Add network adapter 中** 选择正确的网络。
   i. 完成配置并打开虚拟机电源。
2. 继续为集群创建更多计算机器。

12.5.14. 磁盘分区

在大多数情况下，数据分区最初是由安装 RHCOS 而不是安装另一个操作系统来创建的。在这种情况下，OpenShift Container Platform 安装程序应该被允许配置磁盘分区。

但是，在安装 OpenShift Container Platform 节点时，在两种情况下您可能需要覆盖默认分区：

- 创建单独的分区：对于在空磁盘中的 greenfield 安装，您可能想要在分区中添加单独的存储。这只在生成 **/var** 或者一个 **/var** 独立分区的子目录（如 **/var/lib/etcd** ）时被正式支持，但不支持两者。



重要

Kubernetes 只支持两个文件系统分区。如果您在原始配置中添加多个分区，Kubernetes 无法监控所有这些分区。

- 保留现有分区：对于 brownfield 安装，您要在现有节点上重新安装 OpenShift Container Platform，并希望保留从之前的操作系统中安装的数据分区，对于 **coreos-installer** 来说，引导选项和选项都允许您保留现有数据分区。

创建一个独立的 /var 分区

通常情况下，OpenShift Container Platform 的磁盘分区应该留给安装程序。然而，在有些情况下您可能需要在文件系统的一部分中创建独立分区。

OpenShift Container Platform 支持添加单个分区来将存储附加到 **/var** 分区或 **/var** 的子目录。例如：

- **/var/lib/containers**：保存镜像相关的内容，随着更多镜像和容器添加到系统中，它所占用的存储会增加。
- **/var/lib/etcd**：保存您可能希望保持独立的数据，比如 etcd 存储的性能优化。
- **/var**：保存您希望独立保留的数据，用于特定目的（如审计）。

单独存储 **/var** 目录的内容可方便地根据需要对区域扩展存储，并可以在以后重新安装 OpenShift Container Platform 时保持该数据地完整。使用这个方法，您不必再次拉取所有容器，在更新系统时也无法复制大量日志文件。

因为 **/var** 在进行一个全新的 Red Hat Enterprise Linux CoreOS（RHCOS）安装前必需存在，所以这个流程会在 OpenShift Container Platform 安装过程的 **openshift-install** 准备阶段插入的机器配置来设置独立的 **/var** 分区。

流程

1. 创建存放 OpenShift Container Platform 安装文件的目录：

```
$ mkdir $HOME/clusterconfig
```

2. 运行 **openshift-install** 在 **manifest** 和 **openshift** 子目录中创建一组文件。在出现提示时回答系统问题：

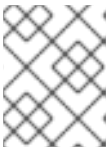
```
$ openshift-install create manifests --dir $HOME/clusterconfig
? SSH Public Key ...
$ ls $HOME/clusterconfig/openshift/
99_kubeadmin-password-secret.yaml
99_openshift-cluster-api_master-machines-0.yaml
99_openshift-cluster-api_master-machines-1.yaml
99_openshift-cluster-api_master-machines-2.yaml
...
```

3. 创建 **MachineConfig** 对象并将其添加到 **openshift** 目录中的一个文件中。例如，把文件命名为 **98-var-partition.yaml**，将磁盘设备名称改为 **worker** 系统中存储设备的名称，并根据情况设置存储大小。这个示例将 **/var** 目录放在一个单独的分区中：

```
apiVersion: machineconfiguration.openshift.io/v1
kind: MachineConfig
metadata:
  labels:
    machineconfiguration.openshift.io/role: worker
  name: 98-var-partition
spec:
  config:
    ignition:
```

```
    version: 3.1.0
  storage:
    disks:
    - device: /dev/<device_name> 1
      partitions:
      - label: var
        startMiB: <partition_start_offset> 2
        sizeMiB: <partition_size> 3
    filesystems:
      - device: /dev/disk/by-partlabel/var
        path: /var
        format: xfs
  systemd:
    units:
      - name: var.mount 4
        enabled: true
        contents: |
          [Unit]
          Before=local-fs.target
          [Mount]
          What=/dev/disk/by-partlabel/var
          Where=/var
          Options=defaults,prjquota 5
          [Install]
          WantedBy=local-fs.target
```

1. 要分区的磁盘的存储设备名称。
2. 当在引导磁盘中添加数据分区时，推荐最少使用 25000MB。root 文件系统会自动重新定义大小使其占据所有可用空间（最多到指定的偏移值）。如果没有指定值，或者指定的值小于推荐的最小值，则生成的 root 文件系统会太小，而在以后进行的 RHCOS 重新安装可能会覆盖数据分区的开始部分。
3. 数据分区的大小（以兆字节为单位）。
4. 挂载单元的名称必须与 **Where=** 指令中指定的目录匹配。例如，对于挂载在 **/var/lib/containers** 上的文件系统，该单元必须命名为 **var-lib-containers.mount**。
5. 对于用于容器存储的文件系统，必须启用 **prjquota** 挂载选项。



注意

在创建独立 **/var** 分区时，如果不同的实例类型没有相同的设备名称，则无法将不同的实例类型用于 worker 节点。

4. 再次运行 **openshift-install**，从 **manifest** 和 **openshift** 子目录中的一组文件创建 Ignition 配置：

```
$ openshift-install create ignition-configs --dir $HOME/clusterconfig
$ ls $HOME/clusterconfig/
auth  bootstrap.ign  master.ign  metadata.json  worker.ign
```

现在，您可以使用 Ignition 配置文件作为 vSphere 安装程序的输入来安装 Red Hat Enterprise Linux CoreOS（RHCOS）系统。

12.5.15. 通过下载二进制文件安装 OpenShift CLI

您需要安装 CLI（**oc**） 来使用命令行界面与 OpenShift Container Platform 进行交互。您可在 Linux、Windows 或 macOS 上安装 **oc**。

重要

如果安装了旧版本的 **oc**，则无法使用 OpenShift Container Platform 4.6 中的所有命令。下载并安装新版本的 **oc**。

12.5.15.1. 在 Linux 上安装 OpenShift CLI

您可以按照以下流程在 Linux 上安装 OpenShift CLI（**oc**）二进制文件。

流程

1. 进入到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 在 **Version** 下拉菜单中选择相应的版本。
3. 单击 **OpenShift v4.6 Linux 客户端** 条目旁边的 **Download Now**，再保存文件。
4. 解包存档：

```
$ tar xvzf <file>
```

5. 把 **oc** 二进制代码放到 **PATH** 中的目录中。
   执行以下命令可以查看当前的 **PATH** 设置：

```
$ echo $PATH
```

安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
$ oc <command>
```

12.5.15.2. 在 Windows 上安装 OpenShift CLI

您可以按照以下流程在 Windows 上安装 OpenShift CLI（**oc**）二进制代码。

流程

1. 进入到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 在 **Version** 下拉菜单中选择相应的版本。
3. 单击 **OpenShift v4.6 Windows 客户端**条目旁边的 **Download Now**，再保存文件。
4. 使用 ZIP 程序解压存档。
5. 把 **oc** 二进制代码放到 **PATH** 中的目录中。
   要查看您的 **PATH**，请打开命令提示窗口并执行以下命令：

```
C:\> path
```

安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
C:\> oc <command>
```

12.5.15.3. 在 macOS 上安装 OpenShift CLI

您可以按照以下流程在 macOS 上安装 OpenShift CLI（**oc**）二进制代码。

流程

1. 进入到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 在 **Version** 下拉菜单中选择相应的版本。
3. 单击 **OpenShift v4.6 MacOSX 客户端**条目旁边的 **Download Now**，再保存文件。
4. 解包和解压存档。
5. 将 **oc** 二进制文件移到 PATH 的目录中。
   要查看您的 **PATH**，打开一个终端窗口并执行以下命令：

   ```
   $ echo $PATH
   ```

安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
$ oc <command>
```

12.5.16. 创建集群

要创建 OpenShift Container Platform 集群，请等待您通过安装程序生成的 Ignition 配置文件所置备的机器上完成 bootstrap 过程。

先决条件

- 为集群创建所需的基础架构。
- 已获得安装程序并为集群生成了 Ignition 配置文件。
- 已使用 Ignition 配置文件为集群创建 RHCOS 机器。
- 您的机器可直接访问互联网，或者可以使用 HTTP 或 HTTPS 代理。

流程

1. 监控 bootstrap 过程：

   ```
   $ ./openshift-install --dir <installation_directory> wait-for bootstrap-complete \ 1
       --log-level=info 2
   ```

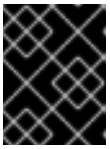
   1. 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。
   2. 要查看不同的安装详情，请指定 **warn**、**debug** 或 **error**，而不要指定 **info**。

输出示例

```
INFO Waiting up to 30m0s for the Kubernetes API at https://api.test.example.com:6443...
INFO API v1.19.0 up
INFO Waiting up to 30m0s for bootstrapping to complete...
INFO It is now safe to remove the bootstrap resources
```

Kubernetes API 服务器提示已在 control plane 机器上完成 bootstrap 时，命令运行成功。

2. bootstrap 过程完成后，请从负载均衡器中删除 bootstrap 机器。



重要

此时您必须从负载均衡器中删除 bootstrap 机器。您还可以删除或重新格式化机器本身。

12.5.17. 使用 CLI 登录到集群

您可以通过导出集群 **kubeconfig** 文件，以默认系统用户身份登录集群。**kubeconfig** 文件包含关于集群的信息，供 CLI 用于将客户端连接到正确集群和 API 服务器。该文件特只适用于一个特定的集群，在 OpenShift Container Platform 安装过程中创建。

先决条件

- 已部署了 OpenShift Container Platform 集群。
- 已安装 **oc** CLI。

流程

1. 导出 **kubeadmin** 凭证：

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

2. 使用导出的配置，验证能否成功运行 **oc** 命令：

```
$ oc whoami
```

输出示例

```
system:admin
```

12.5.18. 批准机器的证书签名请求

将机器添加到集群时，会为您添加的每台机器生成两个待处理证书签名请求（CSR）。您必须确认这些 CSR 已获得批准，或根据需要自行批准。客户端请求必须首先被批准，然后是服务器请求。

先决条件

- 您已将机器添加到集群中。

流程

1. 确认集群可以识别这些机器：

```
$ oc get nodes
```

输出示例

```
NAME      STATUS    ROLES   AGE  VERSION
master-0  Ready     master  63m  v1.19.0
master-1  Ready     master  63m  v1.19.0
master-2  Ready     master  64m  v1.19.0
```

输出将列出您创建的所有机器。

注意

在一些 CSR 被批准前，以上输出可能不包括计算节点（也称为 worker 节点）。

2. 检查待处理的 CSR，并确保可以看到添加到集群中的每台机器都有 **Pending** 或 **Approved** 状态的客户端请求：

```
$ oc get csr
```

输出示例

```
NAME        AGE    REQUESTOR                                                    CONDITION
csr-8b2br   15m    system:serviceaccount:openshift-machine-config-operator:node-
bootstrapper   Pending
csr-8vnps   15m    system:serviceaccount:openshift-machine-config-operator:node-
bootstrapper   Pending
...
```

在本例中，两台机器加入了集群。您可能会在列表中看到更多已批准的 CSR。

3. 如果 CSR 没有获得批准，请在所添加机器的所有待处理 CSR 都处于 **Pending** 状态后，为您的集群机器批准这些 CSR：

注意

由于 CSR 会自动轮转，因此请在将机器添加到集群后一小时内批准您的 CSR。如果没有在一小时内批准，证书将会轮转，每个节点将会存在多个证书。您必须批准所有这些证书。批准客户端 CSR 后， Kubelet 为服务证书创建辅助 CSR，这需要手动批准。然后，如果 Kubelet 请求具有相同参数的新证书，则 **machine-approver** 会自动批准后续服务证书续订请求。

注意

对于在未启用机器 API 的平台中运行的集群，如裸机和其他用户置备的基础架构，必须采用一种方法自动批准 kubelet 提供证书请求（CSR）。如果没有批准请求，则 **oc exec**、**oc rsh** 和 **oc logs** 命令将无法成功，因为 API 服务器连接到 kubelet 时需要服务证书。与 Kubelet 端点联系的任何操作都需要此证书批准。这个方法必须监视新的 CSR，确认 CSR 由 **system:node** 或 **system:admin** 组中的 **node-bootstrapper** 服务帐户提交，并确认节点的身份。

- 若要单独批准，请对每个有效的 CSR 运行以下命令：

```
$ oc adm certificate approve <csr_name> 1
```

1 **<csr_name>** 是当前 CSR 列表中 CSR 的名称。

- 要批准所有待处理的 CSR，请运行以下命令：

```
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}{{"\n"}}
{{end}}{{end}}' | xargs --no-run-if-empty oc adm certificate approve
```

注意

在有些 CSR 被批准前，一些 Operator 可能无法使用。

4. 现在，您的客户端请求已被批准，您必须查看添加到集群中的每台机器的服务器请求：

```
$ oc get csr
```

输出示例

```
NAME        AGE     REQUESTOR                                                   CONDITION
csr-bfd72   5m26s   system:node:ip-10-0-50-126.us-east-2.compute.internal
Pending
csr-c57lv   5m26s   system:node:ip-10-0-95-157.us-east-2.compute.internal
Pending
...
```

5. 如果剩余的 CSR 没有被批准，且处于 **Pending** 状态，请批准集群机器的 CSR：

- 若要单独批准，请对每个有效的 CSR 运行以下命令：

```
$ oc adm certificate approve <csr_name> 1
```

1 **<csr_name>** 是当前 CSR 列表中 CSR 的名称。

- 要批准所有待处理的 CSR，请运行以下命令：

```
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}{{"\n"}}
{{end}}{{end}}' | xargs oc adm certificate approve
```

6. 批准所有客户端和服务器 CSR 后，器将处于 **Ready** 状态。运行以下命令验证：

```
$ oc get nodes
```

输出示例

```
NAME     STATUS   ROLES   AGE  VERSION
master-0 Ready    master  73m  v1.20.0
master-1 Ready    master  73m  v1.20.0
master-2 Ready    master  74m  v1.20.0
worker-0 Ready    worker  11m  v1.20.0
worker-1 Ready    worker  11m  v1.20.0
```

注意

批准服务器 CSR 后可能需要几分钟时间让机器转换为 **Ready** 状态。

其他信息

- 如需有关 CSR 的更多信息，请参阅证书签名请求。

12.5.19. 初始 Operator 配置

在 control plane 初始化后，您必须立即配置一些 Operator 以便它们都可用。

先决条件

- 您的 control plane 已初始化。

流程

1. 观察集群组件上线：

```
$ watch -n5 oc get clusteroperators
```

输出示例

```
NAME                                       VERSION AVAILABLE   PROGRESSING   DEGRADED
SINCE
authentication                             4.6.0   True        False         False      3h56m
cloud-credential                           4.6.0   True        False         False      29h
cluster-autoscaler                         4.6.0   True        False         False      29h
config-operator                            4.6.0   True        False         False      6h39m
console                                    4.6.0   True        False         False      3h59m
csi-snapshot-controller                    4.6.0   True        False         False      4h12m
dns                                        4.6.0   True        False         False      4h15m
etcd                                       4.6.0   True        False         False      29h
image-registry                             4.6.0   True        False         False      3h59m
ingress                                    4.6.0   True        False         False      4h30m
insights                                   4.6.0   True        False         False      29h
kube-apiserver                             4.6.0   True        False         False      29h
kube-controller-manager                    4.6.0   True        False         False      29h
kube-scheduler                             4.6.0   True        False         False      29h
kube-storage-version-migrator              4.6.0   True        False         False      4h2m
machine-api                                4.6.0   True        False         False      29h
```

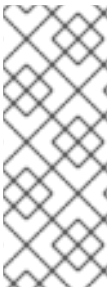
```
machine-approver                           4.6.0   True        False        False      6h34m
machine-config                             4.6.0   True        False        False      3h56m
marketplace                                4.6.0   True        False        False      4h2m
monitoring                                 4.6.0   True        False        False      6h31m
network                                    4.6.0   True        False        False      29h
node-tuning                                4.6.0   True        False        False      4h30m
openshift-apiserver                        4.6.0   True        False        False      3h56m
openshift-controller-manager               4.6.0   True        False        False      4h36m
openshift-samples                          4.6.0   True        False        False      4h30m
operator-lifecycle-manager                 4.6.0   True        False        False      29h
operator-lifecycle-manager-catalog         4.6.0   True        False        False      29h
operator-lifecycle-manager-packageserver   4.6.0   True        False        False      3h59m
service-ca                                 4.6.0   True        False        False      29h
storage                                    4.6.0   True        False        False      4h30m
```

2. 配置不可用的 Operator。

12.5.19.1. 安装过程中删除的镜像 registry

在不提供可共享对象存储的平台上，OpenShift Image Registry Operator bootstraps 本身的状态是 **Removed**。这允许 **openshift-installer** 在这些平台类型上完成安装。

将 **ManagementState** Image Registry Operator 配置从 **Removed** 改为 **Managed**。



注意

Prometheus 控制台提供了一个 **ImageRegistryRemoved** 警报，例如：

"Image Registry has been removed.**ImageStreamTags**, **BuildConfigs** and **DeploymentConfigs** which reference **ImageStreamTags** may not work as expected.Please configure storage and update the config to **Managed** state by editing configs.imageregistry.operator.openshift.io."

12.5.19.2. 镜像 registry 存储配置

对于不提供默认存储的平台，Image Registry Operator 最初将不可用。安装后，您必须配置 registry 使用的存储，这样 Registry Operator 才可用。

示配置生产集群所需的持久性卷的说明。如果适用，显示有关将空目录配置为存储位置的说明，该位置只可用于非生产集群。

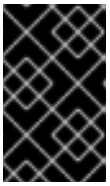
另外还提供了在升级过程中使用 **Recreate** rollout 策略来允许镜像 registry 使用块存储类型的说明。

12.5.19.2.1. 为 VMware vSphere 配置 registry 存储

作为集群管理员，在安装后需要配置 registry 来使用存储。

先决条件

- 具有 Cluster Administrator 权限
- VMware vSphere上有一个集群。
- 为集群置备的持久性存储，如 Red Hat OpenShift Container Storage。



重要

如果您只有一个副本，OpenShift Container Platform 支持对镜像 registry 存储的 **ReadWriteOnce** 访问。要部署支持高可用性的、带有两个或多个副本的镜像 registry，需要 **ReadWriteMany** 访问设置。

- 必须有“100Gi”容量。

重要

测试显示，在 RHEL 中使用 NFS 服务器作为核心服务的存储后端可能会出现问题。这包括 OpenShift Container Registry 和 Quay，Prometheus 用于监控存储，以及 Elasticsearch 用于日志存储。因此，不推荐使用 RHEL NFS 作为 PV 后端用于核心服务。

市场上的其他 NFS 实现可能没有这些问题。如需了解更多与此问题相关的信息，请联络相关的 NFS 厂商。

流程

1. 为了配置 registry 使用存储，需要修改 **configs.imageregistry/cluster** 资源中的 **spec.storage.pvc** 。

注意

使用共享存储时，请查看您的安全设置以防止被外部访问。

2. 验证您没有 registry pod:

```
$ oc get pod -n openshift-image-registry
```

注意

如果存储类型为 **emptyDIR**，则副本数不能超过 **1**。

3. 检查 registry 配置：

```
$ oc edit configs.imageregistry.operator.openshift.io
```

输出示例

```
storage:
  pvc:
    claim: 1
```

1 将 **claim** 字段留空以允许自动创建一个 **image-registry-storage** PVC。

4. 检查 **clusteroperator** 的状态：

```
$ oc get clusteroperator image-registry
```

12.5.19.2.2. 在非生产集群中配置镜像 registry 存储

您必须为 Image Registry Operator 配置存储。对于非生产集群，您可以将镜像 registry 设置为空目录。如果您这样做，重启 registry 后会丢失所有镜像。

流程

- 将镜像 registry 存储设置为空目录：

```
$ oc patch configs.imageregistry.operator.openshift.io cluster --type merge --patch '{"spec":{"storage":{"emptyDir":{}}}}'
```

警告

仅可为非生产集群配置这个选项。

如果在 Image Registry Operator 初始化其组件前运行此命令，**oc patch** 命令会失败并显示以下错误：

```
Error from server (NotFound): configs.imageregistry.operator.openshift.io "cluster" not found
```

等待几分钟，然后再次运行该命令。

12.5.19.2.3. 为 VMware vSphere 配置块 registry 存储

在作为集群管理员升级时，要允许镜像 registry 使用块存储类型，如 vSphere Virtual Machine Disk（VMDK），您可以使用 **Recreate** rollout 策略。

重要

支持块存储卷，但不建议将其用于生产环境中的镜像 registry。在块存储上配置 registry 的安装不具有高可用性，因为 registry 无法拥有多个副本。

流程

1. 要将镜像 registry 存储设置为块存储类型，对 registry 进行补丁，使其使用 **Recreate** rollout 策略，且仅使用 **1** 个副本运行：

```
$ oc patch config.imageregistry.operator.openshift.io/cluster --type=merge -p '{"spec":{"rolloutStrategy":"Recreate","replicas":1}}'
```

2. 为块存储设备置备 PV，并为该卷创建 PVC。请求的块卷使用 ReadWriteOnce（RWO）访问模式。

   a. 创建包含以下内容的 **pvc.yaml** 文件以定义 VMware vSphere **PersistentVolumeClaim**：

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
```

```
  name: image-registry-storage 1
  namespace: openshift-image-registry 2
spec:
  accessModes:
  - ReadWriteOnce 3
  resources:
    requests:
      storage: 100Gi 4
```

1. 代表 **PersistentVolumeClaim** 对象的唯一名称。
2. **PersistentVolumeClaim** 对象的命名空间，即 **openshift-image-registry**。
3. 持久性卷声明的访问模式。使用 **ReadWriteOnce** 时，单个节点可以通过读写权限挂载这个卷。
4. 持久性卷声明的大小。

b. 从文件创建 **PersistentVolumeClaim** 对象：

```
$ oc create -f pvc.yaml -n openshift-image-registry
```

3. 编辑 registry 配置，使其可以正确引用 PVC:

```
$ oc edit config.imageregistry.operator.openshift.io -o yaml
```

输出示例

```
storage:
  pvc:
    claim: 1
```

1. 通过创建自定义 PVC，您可以将 **claim** 字段留空以用于默认自动创建 **image-registry-storage** PVC。

有关配置 registry 存储以便引用正确的 PVC 的说明，请参阅为 vSphere 配置 registry。

12.5.20. 在用户置备的基础架构上完成安装

完成 Operator 配置后，可以在您提供的基础架构上完成集群安装。

先决条件

- 您的 control plane 已初始化。
- 已完成初始 Operator 配置。

流程

1. 使用以下命令确认所有集群组件都已在线：

```
$ watch -n5 oc get clusteroperators
```

输出示例

```
NAME                                       VERSION AVAILABLE   PROGRESSING   DEGRADED
SINCE
authentication                             4.6.0   True        False         False      3h56m
cloud-credential                           4.6.0   True        False         False      29h
cluster-autoscaler                         4.6.0   True        False         False      29h
config-operator                            4.6.0   True        False         False      6h39m
console                                    4.6.0   True        False         False      3h59m
csi-snapshot-controller                    4.6.0   True        False         False      4h12m
dns                                        4.6.0   True        False         False      4h15m
etcd                                       4.6.0   True        False         False      29h
image-registry                             4.6.0   True        False         False      3h59m
ingress                                    4.6.0   True        False         False      4h30m
insights                                   4.6.0   True        False         False      29h
kube-apiserver                             4.6.0   True        False         False      29h
kube-controller-manager                    4.6.0   True        False         False      29h
kube-scheduler                             4.6.0   True        False         False      29h
kube-storage-version-migrator              4.6.0   True        False         False      4h2m
machine-api                                4.6.0   True        False         False      29h
machine-approver                           4.6.0   True        False         False      6h34m
machine-config                             4.6.0   True        False         False      3h56m
marketplace                                4.6.0   True        False         False      4h2m
monitoring                                 4.6.0   True        False         False      6h31m
network                                    4.6.0   True        False         False      29h
node-tuning                                4.6.0   True        False         False      4h30m
openshift-apiserver                        4.6.0   True        False         False      3h56m
openshift-controller-manager               4.6.0   True        False         False      4h36m
openshift-samples                          4.6.0   True        False         False      4h30m
operator-lifecycle-manager                 4.6.0   True        False         False      29h
operator-lifecycle-manager-catalog         4.6.0   True        False         False      29h
operator-lifecycle-manager-packageserver   4.6.0   True        False         False      3h59m
service-ca                                 4.6.0   True        False         False      29h
storage                                    4.6.0   True        False         False      4h30m
```

或者，通过以下命令，如果所有集群都可用您会接到通知。它还检索并显示凭证：

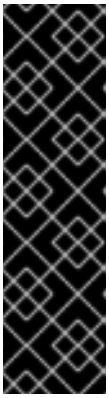
```
$ ./openshift-install --dir <installation_directory> wait-for install-complete 1
```

1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

输出示例

```
INFO Waiting up to 30m0s for the cluster to initialize...
```

Cluster Version Operator 完成从 Kubernetes API 服务器部署 OpenShift Container Platform 集群时，命令运行成功。



重要

- 安装程序生成的 Ignition 配置文件包含在 24 小时后过期的证书，然后在过期时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外情况是，您需要手动批准待处理的 **node-bootstrapper** 证书签名请求（CSR）来恢复 kubelet 证书。如需更多信息，请参阅*从过期的 control plane 证书中恢复*的文档。

- 建议您在生成 12 小时后使用 Ignition 配置文件，因为集群安装后 24 小时证书从 16 小时轮转至 22 小时。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中运行证书更新时避免安装失败。

2. 确认 Kubernetes API 服务器正在与 pod 通信。

   a. 要查看所有 pod 的列表，请使用以下命令：

   ```
   $ oc get pods --all-namespaces
   ```

   输出示例

   ```
   NAMESPACE                       NAME                                         READY   STATUS
   RESTARTS   AGE
   openshift-apiserver-operator    openshift-apiserver-operator-85cb746d55-zqhs8   1/1
   Running   1          9m
   openshift-apiserver             apiserver-67b9g                               1/1     Running   0
   3m
   openshift-apiserver             apiserver-ljcmx                               1/1     Running   0
   1m
   openshift-apiserver             apiserver-z25h4                               1/1     Running   0
   2m
   openshift-authentication-operator authentication-operator-69d5d8bf84-vh2n8     1/1
   Running   0          5m
   ...
   ```

   b. 使用以下命令，查看上一命令的输出中所列 pod 的日志：

   ```
   $ oc logs <pod_name> -n <namespace>
   ```
   (1)

   (1) 指定 pod 名称和命名空间，如上一命令的输出中所示。

   如果 pod 日志显示，Kubernetes API 服务器可以与集群机器通信。

您可以按照将计算机器添加到 vSphere 的内容，在集群安装完成后添加额外的计算机器。

12.5.21. 备份 VMware vSphere 卷

OpenShift Container Platform 将新卷作为独立持久性磁盘置备，以便在集群中的任何节点上自由附加和分离卷。因此，无法备份使用快照的卷，也无法从快照中恢复卷。如需更多信息，请参阅 快照限制。

流程

要创建持久性卷的备份：

1. 停止使用持久性卷的应用程序。

2. 克隆持久性卷。
3. 重启应用程序。
4. 创建克隆的卷的备份。
5. 删除克隆的卷。

12.5.22. OpenShift Container Platform 的 Telemetry 访问

在 OpenShift Container Platform 4.6 中，默认运行的 Telemetry 服务提供有关集群健康状况和成功更新的指标，需要访问互联网。如果您的集群连接到互联网，Telemetry 会自动运行，而且集群会注册到 OpenShift Cluster Manager。

确认 OpenShift Cluster Manager 清单正确后，可以由 Telemetry 自动维护，也可以使用 OpenShift Cluster Manager 手动维护，使用订阅监控来跟踪帐户或多集群级别的 OpenShift Container Platform 订阅。

其他资源

- 有关 Telemetry 服务的更多信息，请参阅关于远程健康监控。

12.5.23. 后续步骤

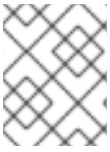
- 自定义集群。
- 如果需要，您可以选择不使用远程健康报告。
- 设置 registry 并配置 registry 存储。

12.6. 使用用户置备的基础架构和网络自定义在 VMC 上安装集群

在 OpenShift Container Platform 版本 4.6 中，您可以使用带有自定义的网络配置选项置备的基础架构 VMware vSphere 实例上安装集群，方法是将其部署到 VMware Cloud (VMC) on AWS。

为 OpenShift Container Platform 部署配置了 VMC 环境后，您要使用堡垒管理主机中的 OpenShift Container Platform 安装程序，并位于 VMC 环境中。安装程序和 control plane 可以自动部署和管理 OpenShift Container Platform 集群所需的资源。

通过自定义网络配置，您的集群可以与环境中现有的 IP 地址分配共存，并与现有的 VXLAN 配置集成。大部分网络配置参数必须在安装过程中设置，只有 **kubeProxy** 配置参数可以在运行的集群中修改。

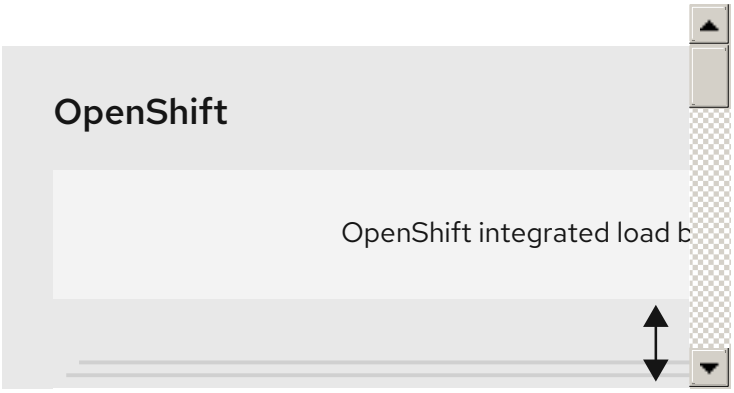


注意

OpenShift Container Platform 支持将集群部署到单个 VMware vCenter 中。不支持在多个 vCenter 上使用机器/机器集部署集群。

12.6.1. 为 vSphere 设置 VMC

您可以在 AWS 托管的 vSphere 集群上安装 OpenShift Container Platform（VMC），以便启用在混合云内部和内部管理应用程序。



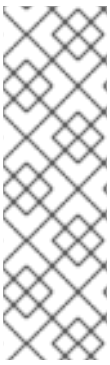
在 VMware vSphere 上安装 OpenShift Container Platform 之前，您必须在 VMC 环境中配置多个选项。确定您的 VMC 环境有以下先决条件：

- 创建非专用的、启用了 DHCP 的、NSX-T 网络片段和子网。其他虚拟机（VM）可以托管在子网上，但至少有 8 个 IP 地址可用于 OpenShift Container Platform 部署。
- 配置以下防火墙规则：
  - OpenShift Container Platform 计算网络和互联网间的 ANY:ANY 防火墙规则。节点和应用程序用于下载容器镜像。
  - 在安装主机和软件定义的数据中心（SDDC）管理网络之间的端口 443 的 ANY:ANY 防火墙规则。这可让您在部署过程中上传 Red Hat Enterprise Linux CoreOS（RHCOS）OVA。
  - OpenShift Container Platform 计算网络和 vCenter 间的 HTTPS 防火墙规则。此连接允许 OpenShift Container Platform 与 vCenter 通信，以置备和管理节点、持久性卷声明（PVC）和其他资源。
- 您必须具有以下信息才能部署 OpenShift Container Platform:
  - OpenShift Container Platform 集群名称，如 **vmc-prod-1**。
  - 基本 DNS 名称，如 **companyname.com**。
  - 如果没有使用默认设置，则必须识别 Pod 网络 CIDR 和服务网络 CIDR，默认设置为 **10.128.0.0/14** 和 **172.30.0.0/16**。这些 CIDR 用于 pod 到 pod 和 pod 到服务的通信，且无法在外部访问，但不得与机构中的现有子网重叠。
  - 以下 vCenter 信息：
    - vCenter 主机名、用户名和密码
    - 数据中心名称，如 **SDDC-Datacenter**
    - 集群名称，如 **Cluster-1**
    - 网络名称
    - 数据存储名称，如 **WorkloadDatastore**

注意

建议在集群安装完成后将 vSphere 集群移到 VMC **Compute-ResourcePool** 资源池。

- 基于 Linux 的主机作为堡垒部署到 VMC。
  - 堡垒主机可以是 Red Hat Enterprise Linux（RHEL）或其他基于 Linux 的主机，它必须具有互联网连接，并可以将 OVA 上传到 ESXi 主机。
  - 将 OpenShift CLI 工具下载并安装到堡垒主机。
    - **openshift-install** 安装程序
    - OpenShift CLI（**oc**）工具



注意

您不能将 VMware NSX Container Plugin 用于 Kubernetes（NCP），NCP 不使用 NSX 作为 OpenShift SDN。目前 VMC 提供的 NSX 版本与 OpenShift Container Platform 认证的 NCP 版本不兼容。

但是，NSX DHCP 服务会通过全堆栈自动 OpenShift Container Platform 部署来管理虚拟机 IP 管理，节点可以手动或自动置备，可通过 Machine API 与 vSphere 集成。另外，还创建 NSX 防火墙规则，以便启用 OpenShift Container Platform 集群以及堡垒主机和 VMC vSphere 主机间的访问。

12.6.1.1. VMC Sizer 工具

AWS 上的 VMware Cloud 基于 AWS 裸机基础架构构建，这与运行 AWS 原生服务的裸机基础架构相同。当在 AWS 软件定义的数据中心（SDDC）上部署 VMware 云时，您要将这些物理服务器节点以单一租户方式运行 VMware ESXi hypervisor。这意味着其他使用 VMC 的用户无法访问物理基础结构。务必要考虑托管您的虚拟基础架构所需的物理主机数量。

要确定这一点，VMware 在 AWS Sizer 上提供 VMC。使用这个工具，您可以定义要在 VMC 上托管的资源：

- 工作负载类型
- 虚拟机总数
- 规格信息，例如：
  - 存储要求
  - vCPUs
  - vRAM
  - 过量使用比例

通过这些信息，sizer 工具可以根据 VMware 最佳实践生成报告，并推荐集群配置和您需要的主机数量。

12.6.2. vSphere 先决条件

- 置备块 registry 存储。如需有关持久性存储的更多信息，请参阅了解持久性存储。
- 查看有关 OpenShift Container Platform 安装和更新流程的详细信息。
- 如果使用防火墙，您必须 将其配置为访问 Red Hat Insights。

12.6.3. OpenShift Container Platform 的互联网访问

在 OpenShift Container Platform 4.6 中，您需要访问互联网来安装集群。

您必须具有以下互联网访问权限：

- 访问 OpenShift Cluster Manager 以下载安装程序并执行订阅管理。如果集群可以访问互联网，并且没有禁用 Telemetry，该服务会自动授权您的集群。
- 访问 Quay.io，以获取安装集群所需的软件包。
- 获取执行集群更新所需的软件包。

重要

如果您的集群无法直接访问互联网，则可以在置备的某些类基础架构上执行受限网络安装。在此过程中，您要下载所需的内容，并使用它在镜像 registry（mirror registry） 中填充安装集群并生成安装程序所需的软件包。对于某些安装类型，集群要安装到的环境不需要访问互联网。在更新集群之前，要更新 registry 镜像系统中的内容。

12.6.4. VMware vSphere 基础架构要求

您必须在满足您使用的组件要求的 VMware vSphere 版本 6 或 7 实例上安装 OpenShift Container Platform 集群。

表 12.49. VMware 组件支持的最低 vSphere 版本

组件	最低支持版本	描述
虚拟机监控程序	vSphere 6.5 及之后的版本 13	此版本是 Red Hat Enterprise Linux CoreOS(RHCOS)支持的最低版本。请查看 Red Hat Enterprise Linux 8 支持的管理程序列表。
使用 in-tree 驱动程序存储	vSphere 6.5 及之后的版本	此插件使用 OpenShift Container Platform 中包含的 vSphere 的树内存储驱动程序创建 vSphere 存储。

如果您使用 vSphere 版本 6.5 实例，请在安装 OpenShift Container Platform 前考虑升级到 6.7U3 或 7.0。

重要

您必须确保在安装 OpenShift Container Platform 前同步 ESXi 主机上的时间。请参阅 VMware 文档中的编辑主机时间配置。

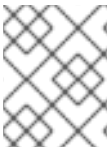
12.6.5. 具有用户置备基础架构的集群的机器要求

对于含有用户置备的基础架构的集群，您必须部署所有所需的机器。

12.6.5.1. 所需的机器

最小的 OpenShift Container Platform 集群需要下列主机：

- 一个临时 bootstrap 机器
- 三台 control plane 或 master 机器
- 至少两台计算机器，也称为 worker 机器。



注意

集群要求 bootstrap 机器在三台 control plane 机器上部署 OpenShift Container Platform 集群。您可在安装集群后删除 bootstrap 机器。

重要

要保持集群的高可用性，请将独立的物理主机用于这些集群机器。

bootstrap 和 control plane 机器必须使用 Red Hat Enterprise Linux CoreOS (RHCOS) 作为操作系统。但是，计算机器可以在 Red Hat Enterprise Linux CoreOS(RHCOS)或 Red Hat Enterprise Linux(RHEL)7.9 之间进行选择。

请注意，RHCOS 基于 Red Hat Enterprise Linux（RHEL） 8，并继承其所有硬件认证和要求。请查看Red Hat Enterprise Linux 技术功能及限制 。

12.6.5.2. 网络连接要求

所有 Red Hat Enterprise Linux CoreOS (RHCOS) 机器在启动过程中需要 **initramfs** 中的网络从 Machine Config Server 获取 Ignition 配置文件。在初次启动过程中，需要一个 DHCP 服务器或设置了静态 IP 地址来建立网络连接，以下载它们的 Ignition 配置文件。另外，集群中的每个 OpenShift Container Platform 节点都必须有权访问网络时间协议（NTP）服务器。如果 DHCP 服务器提供 NTP 服务器信息，Red Hat Enterprise Linux CoreOS（RHCOS）机器上的 chrony 时间服务会读取信息，并可与 NTP 服务器同步时钟。

12.6.5.3. 最低资源要求

每台集群机器都必须满足以下最低要求：

表 12.50. 最低资源要求

机器	操作系统	vCPU [1]	虚拟内存	存储	IOPS [2]
bootstrap	RHCOS	4	16 GB	100 GB	300
Control plane	RHCOS	4	16 GB	100 GB	300
Compute	RHCOS 或 RHEL 7.9	2	8 GB	100 GB	300

1. 当未启用并发多线程 (SMT) 或超线程时，一个 vCPU 相当于一个物理内核。启用后，使用以下公式来计算对应的比例：（每个内核数的线程）× sockets = vCPU。

2. OpenShift Container Platform 和 Kubernetes 对磁盘性能非常敏感，建议使用更快的存储速度，特别是 control plane 节点上需要 10 ms p99 fsync 持续时间的 etcd。请注意，在许多云平台上，存储大小和 IOPS 可一起扩展，因此您可能需要过度分配存储卷来获取足够的性能。

12.6.5.4. 证书签名请求管理

在使用您置备的基础架构时，集群只能有限地访问自动机器管理，因此您必须提供一种在安装后批准集群证书签名请求 (CSR) 的机制。**kube-controller-manager** 只能批准 kubelet 客户端 CSR。 **machine-approver** 无法保证使用 kubelet 凭证请求的提供证书的有效性，因为它不能确认是正确的机器发出了该请求。您必须决定并实施一种方法，以验证 kubelet 提供证书请求的有效性并进行批准。

12.6.6. 创建用户置备的基础架构

在部署采用用户置备的基础架构的 OpenShift Container Platform 集群前，您必须创建底层基础架构。

先决条件

- 在为集群创建支持基础架构之前，请参阅OpenShift Container Platform 4.x Tested Integrations页。

流程

1. 在每个节点上配置 DHCP 或设置静态 IP 地址。
2. 提供所需的负载均衡器。
3. 配置机器的端口。
4. 配置 DNS。
5. 确保网络可以正常工作。

12.6.6.1. 用户置备的基础架构对网络的要求

所有 Red Hat Enterprise Linux CoreOS（RHCOS）机器在启动过程中需要 **initramfs** 中的网络从机器配置服务器获取 Ignition 配置。

在初次启动过程中，需要一个 DHCP 服务器或集群中的每个机器都设置了静态 IP 地址来建立网络连接，以下载它们的 Ignition 配置文件。

建议您使用 DHCP 服务器为集群进行长期机器管理。确保 DHCP 服务器已配置为向集群机器提供持久 IP 地址和主机名。

Kubernetes API 服务器必须能够解析集群机器的节点名称。如果 API 服务器和 worker 节点位于不同的区域中，您可以配置默认 DNS 搜索区域，以便 API 服务器能够解析节点名称。另一种支持的方法是始终在节点对象和所有 DNS 请求中使用完全限定域名来指代主机。

您必须配置机器间的网络连接，以便集群组件进行通信。每台机器都必须能够解析集群中所有其他机器的主机名。

表 12.51. 所有机器到所有机器

协议	端口	描述
ICMP	N/A	网络可访问性测试

协议	端口	描述
TCP	**1936**	指标
	9000-9999	主机级别的服务，包括端口 **9100-9101** 上的节点导出器和端口 **9099** 上的 Cluster Version Operator。
	10250-10259	Kubernetes 保留的默认端口
	10256	openshift-sdn
UDP	**4789**	VXLAN 和 Geneve
	6081	VXLAN 和 Geneve
	9000-9999	主机级别的服务，包括端口 **9100-9101** 上的节点导出器。
TCP/UDP	**30000-32767**	Kubernetes 节点端口

表 12.52. 要通过控制平面的所有机器

协议	端口	描述
TCP	**6443**	Kubernetes API

表 12.53. control plane 机器到 control plane 机器

协议	端口	描述
TCP	**2379-2380**	etcd 服务器和对等端口

网络拓扑要求

您为集群置备的基础架构必须满足下列网络拓扑要求。

重要

OpenShift Container Platform 要求所有节点都能访问互联网，以便为平台容器提取镜像并向红帽提供遥测数据。

负载均衡器

在安装 OpenShift Container Platform 前，您必须置备两个满足以下要求的负载均衡器：

1. **API 负载均衡器**：提供一个通用端点，供用户（包括人和机器）与平台交互和配置。配置以下条件：

- 只适用于第 4 层负载均衡。这可被称为 Raw TCP、SSL Passthrough 或者 SSL 桥接模式。如果使用 SSL Bridge 模式，必须为 API 路由启用 Server Name Indication（SNI）。
- 无状态负载平衡算法。这些选项根据负载均衡器的实现而有所不同。

重要

不要为 API 负载均衡器配置会话持久性。

在负载均衡器的前端和后台配置以下端口：

表 12.54. API 负载均衡器

端口	后端机器（池成员）	内部	外部	描述
6443	Bootstrap 和 control plane.bootstrap 机器初始化集群 control plane 后，您要从负载均衡器中删除 bootstrap 机器。您必须为 API 服务器健康检查探测配置 **/readyz** 端点。	X	X	Kubernetes API 服务器
22623	Bootstrap 和 control plane.bootstrap 机器初始化集群 control plane 后，您要从负载均衡器中删除 bootstrap 机器。	X		机器配置服务器

注意

负载均衡器必须配置为，从 API 服务器关闭 **/readyz** 端点到从池中删除 API 服务器实例时最多需要 30 秒。在 **/readyz** 返回错误或处于健康状态后的时间范围内，端点必须被删除或添加。每 5 秒或 10 秒探测一次，有两个成功请求处于健康状态，三个成为不健康的请求经过测试。

2. **应用程序入口负载均衡器**:提供来自集群外部的应用程序流量流量的 Ingress 点。配置以下条件：
   - 只适用于第 4 层负载均衡。这可被称为 Raw TCP、SSL Passthrough 或者 SSL 桥接模式。如果使用 SSL Bridge 模式，您必须为 Ingress 路由启用Server Name Indication（SNI）。
   - 建议根据可用选项以及平台上托管的应用程序类型，使用基于连接的或者基于会话的持久性。

在负载均衡器的前端和后台配置以下端口：

表 12.55. 应用程序入口负载均衡器

端口	后端机器（池成员）	内部	外部	描述
443	默认运行入口路由器 Pod、计算或 worker 的机器。	X	X	HTTPS 流量
80	默认运行入口路由器 Pod、计算或 worker 的机器。	X	X	HTTP 流量

提示

如果负载均衡器可以看到客户端的真实 IP 地址，启用基于 IP 的会话持久性可提高使用端到端 TLS 加密的应用程序的性能。

注意

OpenShift Container Platform 集群需要正确配置入口路由器。control plane 初始化后，您必须配置入口路由器。

NTP 配置

OpenShift Container Platform 集群默认配置为使用公共网络时间协议（NTP）服务器。如果要使用本地企业 NTP 服务器，或者集群部署在断开连接的网络中，您可以将集群配置为使用特定的时间服务器。如需更多信息，请参阅*配置 chrony 时间服务*的文档。

如果 DHCP 服务器提供 NTP 服务器信息，Red Hat Enterprise Linux CoreOS（RHCOS）机器上的 chrony 时间服务会读取信息，并可与 NTP 服务器同步时钟。

12.6.6.2. 用户置备 DNS 要求

DNS 用于名称解析和反向名称解析。DNS A/AAAA 或 CNAME 记录用于名称解析，PTR 记录用于反向解析名称。反向记录很重要，因为 Red Hat Enterprise Linux CoreOS（RHCOS）使用反向记录为所有节点设置主机名。另外，反向记录用于生成 OpenShift Container Platform 需要操作的证书签名请求（CSR）。

采用用户置备的基础架构的 OpenShift Container Platform 集群需要以下 DNS 记录。在每一记录中，**<cluster_name>** 是集群名称，**<base_domain>** 则是您在 **install-config.yaml** 文件中指定的集群基域。完整的 DNS 记录采用如下格式: **<component>.<cluster_name>.<base_domain>.**。

表 12.56. 所需的 DNS 记录

组件	记录	描述
Kubernetes API	**api.<cluster_name>.<base_domain>.**	添加 DNS A/AAAA 或 CNAME 记录，以及 DNS PTR 记录，以识别 control plane 机器的负载均衡器。这些记录必须由集群外的客户端以及集群中的所有节点解析。
	api-int.<cluster_name>.<base_domain>.	添加 DNS A/AAAA 或 CNAME 记录，以及 DNS PTR 记录，以识别 control plane 机器的负载均衡器。这些记录必须可以从集群中的所有节点解析。  **重要** API 服务器必须能够根据在 Kubernetes 中记录的主机名解析 worker 节点。如果 API 服务器无法解析节点名称，则代理的 API 调用会失败，且您无法从 pod 检索日志。
Routes	***.apps.<cluster_name>.<base_domain>.**	添加通配符 DNS A/AAAA 或 CNAME 记录，指向以运行入口路由器 Pod 的机器（默认为 worker 节点）为目标的负载均衡器。这些记录必须由集群外的客户端以及集群中的所有节点解析。

组件	记录	描述
bootstrap	**bootstrap.<cluster_name>.<base_domain>.**	添加 DNS A/AAAA 或 CNAME 记录，以及 DNS PTR 记录来识别 bootstrap 机器。这些记录必须由集群中的节点解析。
Master 主机	**<master><n>.<cluster_name>.<base_domain>.**	DNS A/AAAA 或 CNAME 记录，以识别 control plane 节点（也称为 master 节点）的每台机器。这些记录必须由集群中的节点解析。
Worker 主机	**<worker><n>.<cluster_name>.<base_domain>.**	添加 DNS A/AAAA 或 CNAME 记录，以识别 worker 节点的每台机器。这些记录必须由集群中的节点解析。

提示

您可以使用 **nslookup <hostname>** 命令来验证名称解析。您可以使用 **dig -x <ip_address>** 命令来验证 PTR 记录的反向名称解析。

下面的 BIND 区文件的例子展示了关于名字解析的 A 记录的例子。这个示例的目的是显示所需的记录。这个示例不是为选择一个名称解析服务提供建议。

例 12.11. DNS 区数据库示例

```
$TTL 1W
@ IN SOA ns1.example.com. root (
   2019070700 ; serial
   3H  ; refresh (3 hours)
   30M  ; retry (30 minutes)
   2W  ; expiry (2 weeks)
   1W )  ; minimum (1 week)
 IN NS ns1.example.com.
 IN MX 10 smtp.example.com.
;
;
ns1 IN A 192.168.1.5
smtp IN A 192.168.1.5
;
helper IN A 192.168.1.5
helper.ocp4 IN A 192.168.1.5
;
; The api identifies the IP of your load balancer.
api.ocp4  IN A 192.168.1.5
api-int.ocp4  IN A 192.168.1.5
;
; The wildcard also identifies the load balancer.
*.apps.ocp4  IN A 192.168.1.5
;
; Create an entry for the bootstrap host.
bootstrap.ocp4 IN A 192.168.1.96
;
; Create entries for the master hosts.
```

```
master0.ocp4  IN A 192.168.1.97
master1.ocp4  IN A 192.168.1.98
master2.ocp4  IN A 192.168.1.99
;
; Create entries for the worker hosts.
worker0.ocp4  IN A 192.168.1.11
worker1.ocp4  IN A 192.168.1.7
;
;EOF
```

下面的 BIND 区文件示例显示了反向名字解析的 PTR 记录示例。

例 12.12. 反向记录的 DNS 区数据库示例

```
$TTL 1W
@ IN SOA ns1.example.com. root (
   2019070700 ; serial
   3H  ; refresh (3 hours)
   30M  ; retry (30 minutes)
   2W  ; expiry (2 weeks)
   1W )  ; minimum (1 week)
 IN NS ns1.example.com.
;
; The syntax is "last octet" and the host must have an FQDN
; with a trailing dot.
97 IN PTR master0.ocp4.example.com.
98 IN PTR master1.ocp4.example.com.
99 IN PTR master2.ocp4.example.com.
;
96 IN PTR bootstrap.ocp4.example.com.
;
5 IN PTR api.ocp4.example.com.
5 IN PTR api-int.ocp4.example.com.
;
11 IN PTR worker0.ocp4.example.com.
7 IN PTR worker1.ocp4.example.com.
;
;EOF
```

12.6.7. 生成 SSH 私钥并将其添加到代理中

如果要在集群上执行安装调试或灾难恢复，则必须为 **ssh-agent** 和安装程序提供 SSH 密钥。您可以使用此密钥访问公共集群中的 bootstrap 机器来排除安装问题。

注意

在生产环境中，您需要进行灾难恢复和调试。

您可以使用此密钥以 **core** 用户身份通过 SSH 连接到 master 节点。在部署集群时，此密钥会添加到 **core** 用户的 **~/.ssh/authorized_keys** 列表中。

注意

您必须使用一个本地密钥，而不要使用在特定平台上配置的密钥，如 AWS 密钥对。

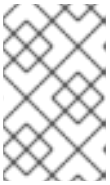
流程

1. 如果还没有为计算机上免密码身份验证而配置的 SSH 密钥，请创建一个。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ ssh-keygen -t ed25519 -N '' \
  -f <path>/<file_name> 1
```

1 指定新 SSH 密钥的路径和文件名，如 **~/.ssh/id_rsa**。如果您已有密钥对，请确保您的公钥位于 **~/.ssh** 目录中。

运行此命令会在指定的位置生成不需要密码的 SSH 密钥。



注意

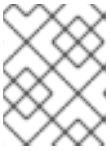
如果您计划在 **x86_64** 架构中安装使用 FIPS 验证的/Modules in Process 加密库的 OpenShift Container Platform 集群，不要创建使用 **ed25519** 算法的密钥。反之，创建一个使用 **rsa** 或 **ecdsa** 算法的密钥。

2. 作为后台任务启动 **ssh-agent** 进程：

```
$ eval "$(ssh-agent -s)"
```

输出示例

```
Agent pid 31874
```



注意

如果您的集群采用 FIPS 模式，则只使用 FIPS 兼容算法来生成 SSH 密钥。密钥必须是 RSA 或 ECDSA。

3. 将 SSH 私钥添加到 **ssh-agent**：

```
$ ssh-add <path>/<file_name> 1
```

输出示例

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

1 指定 SSH 私钥的路径和文件名，如 **~/.ssh/id_rsa**

后续步骤

- 在安装 OpenShift Container Platform 时，为安装程序提供 SSH 公钥。

12.6.8. 获取安装程序

在安装 OpenShift Container Platform 之前，将安装文件下载到本地计算机上。

先决条件

- 运行 Linux 或 macOS 的计算机，本地磁盘空间为 500 MB

流程

1. 访问 OpenShift Cluster Manager 站点的 Infrastructure Provider 页面。如果您有红帽帐号，请使用自己的凭证登录。如果没有，请创建一个帐户。
2. 选择您的基础架构供应商。
3. 进入适用于您的安装类型的页面，下载您的操作系统的安装程序，并将文件放在要保存安装配置文件的目录中。。

重要

安装程序会在用来安装集群的计算机上创建若干文件。在完成集群安装后，您必须保留安装程序和安装程序所创建的文件。这两个文件都需要删除集群。

重要

删除安装程序创建的文件不会删除您的集群，即使集群在安装过程中失败也是如此。要删除集群，为特定云供应商完成 OpenShift Container Platform 卸载流程。

4. 提取安装程序。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ tar xvf openshift-install-linux.tar.gz
```

5. 从 Red Hat OpenShift Cluster Manager 下载安装 pull secret 。通过此 pull secret，您可以进行所含授权机构提供的服务的身份验证，这些服务包括为 OpenShift Container Platform 组件提供容器镜像的 Quay.io。

12.6.9. 手动创建安装配置文件

对于使用用户置备的基础架构的 OpenShift Container Platform 安装，您必须手动生成安装配置文件。

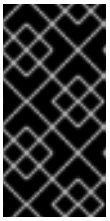
先决条件

- 获取 OpenShift Container Platform 安装程序和集群的访问令牌。

流程

1. 创建用来存储您所需的安装资产的安装目录：

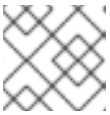
```
$ mkdir <installation_directory>
```



重要

您必须创建目录。一些安装信息，如 bootstrap X.509 证书，有较短的过期间隔，因此不要重复使用安装目录。如果要重复使用另一个集群安装中的个别文件，可以将其复制到您的目录中。但是，一些安装数据的文件名可能会在发行版本之间有所改变。从 OpenShift Container Platform 老版本中复制安装文件时要格外小心。

2. 自定义以下 **install-config.yaml** 文件模板，并将它保存到 **<installation_directory>** 中。



注意

此配置文件必须命名为 **install-config.yaml**。

3. 备份 **install-config.yaml** 文件，以便用于安装多个集群。

重要

install-config.yaml 文件会在安装过程的下一步骤中消耗掉。现在必须备份它。

12.6.9.1. VMware vSphere install-config.yaml 文件示例

您可以自定义 **install-config.yaml** 文件，以指定有关 OpenShift Container Platform 集群平台的更多信息，或修改所需参数的值。

```
apiVersion: v1
baseDomain: example.com (1)
compute:
- hyperthreading: Enabled (2) (3)
  name: worker
  replicas: 0 (4)
controlPlane:
  hyperthreading: Enabled (5) (6)
  name: master
  replicas: 3 (7)
metadata:
  name: test (8)
platform:
  vsphere:
    vcenter: your.vcenter.server (9)
    username: username (10)
    password: password (11)
    datacenter: datacenter (12)
    defaultDatastore: datastore (13)
    folder: "/<datacenter_name>/vm/<folder_name>/<subfolder_name>" (14)
fips: false (15)
pullSecret: '{"auths": ...}' (16)
sshKey: 'ssh-ed25519 AAAA...' (17)
```

(1) 集群的基域。所有 DNS 记录都必须是这个基域的子域，并包含集群名称。

(2) (5) **controlPlane** 部分是一个单映射，但 compute 部分是一系列映射。为满足不同数据结构的要求，**compute** 部分的第一行必须以连字符 **-** 开头，**controlPlane** 部分的第一行则不可以连字符开

头。虽然这两个部分目前都定义单个机器池，但未来的 OpenShift Container Platform 版本可能会支持在安装过程中定义多个计算池。只使用一个 control plane 池。

3 6 是否要启用或禁用并发多线程或**超线程**。默认情况下，启用并发多线程以提高机器内核的性能。您可以通过将参数值设为 **Disabled** 来禁用。如果您在某些集群机器上禁用并发多线程，则必须在所有集群机器上禁用。

重要

如果禁用并发多线程，请确保在容量规划时考虑到机器性能可能会显著降低的问题。如果您禁用并发多线程，则计算机必须至少使用 8 个 CPU 和 32GB RAM。

4 **replicas** 参数的值必须设置为 **0**。此参数控制集群为您创建和管理的 worker 数量，使用用户置备的基础架构时集群不会执行这些功能。在完成 OpenShift Container Platform 安装前，您必须手动为集群部署 worker 机器。

7 您添加到集群的 control plane 机器数量。由于集群将这个值用作集群中 etcd 端点的数量，因此该值必须与您部署的 control plane 机器数量匹配。

8 您在 DNS 记录中指定的集群名称。

9 vCenter 服务器的完全限定主机名或 IP 地址。

10 用于访问服务器的用户名。此用户必须至少具有 vSphere 中 静态或动态持久性卷置备 所需的角色和权限。

11 与 vSphere 用户关联的密码。

12 vSphere 数据中心。

13 要使用的默认 vSphere 数据存储。

14 可选：对于安装程序置备的基础架构，安装程序创建虚拟机的现有文件夹的绝对路径，如 **/<datacenter_name>/vm/<folder_name>/<subfolder_name>**。如果没有提供这个值，安装程序会在数据中心虚拟机文件夹中创建一个顶层文件夹，其名称为基础架构 ID。如果您为集群提供基础架构,请省略此参数。

15 是否启用或禁用 FIPS 模式。默认情况下不启用 FIPS 模式。如果启用了 FIPS 模式，运行 OpenShift Container Platform 的 Red Hat Enterprise Linux CoreOS (RHCOS) 机器会绕过默认的 Kubernetes 加密套件，并使用由 RHCOS 提供的加密模块。

重要

只有在 **x86_64** 架构中的 OpenShift Container Platform 部署支持 FIPS 验证的/Modules in Process 加密库。

16 从 OpenShift Cluster Manager 获取的 pull secret。通过此 pull secret，您可以进行所含授权机构提供的服务的身份验证，这些服务包括为 OpenShift Container Platform 组件提供容器镜像的 Quay.io。

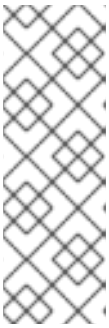
17 Red Hat Enterprise Linux CoreOS (RHCOS) 中 **core** 用户的默认 SSH 密钥的公钥部分。

12.6.9.2. 在安装过程中配置集群范围代理

生产环境可能会拒绝直接访问互联网，而是提供 HTTP 或 HTTPS 代理。您可以通过在 **install-config.yaml** 文件中配置代理设置，将新的 OpenShift Container Platform 集群配置为使用代理。

先决条件

- 您有一个现有的 **install-config.yaml** 文件。
- 您检查了集群需要访问的站点，并决定是否需要绕过代理。默认情况下代理所有集群出口流量，包括对托管云供应商 API 的调用。您需要将站点添加到 **Proxy** 对象的 **spec.noProxy** 字段来绕过代理。



注意

Proxy 对象 **status.noProxy** 字段使用安装配置中的 **networking.machineNetwork[].cidr**、**networking.clusterNetwork[].cidr** 和 **networking.serviceNetwork[]** 字段的值填充。

对于在 Amazon Web Services(AWS)、Google Cloud Platform(GCP)、Microsoft Azure 和 Red Hat OpenStack Platform(RHOSP)上安装, **Proxy** 对象 **status.noProxy** 字段也会使用实例元数据端点填充(**169.254.169.254**)。

流程

1. 编辑 **install-config.yaml** 文件并添加代理设置。例如：

```
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: example.com 3
additionalTrustBundle: | 4
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
...
```

1. 用于创建集群外 HTTP 连接的代理 URL。URL 必须是 **http**。
2. 用于创建集群外 HTTPS 连接的代理 URL。
3. 要排除在代理中的目标域名、IP 地址或其他网络 CIDR 的逗号分隔列表。在域前面加 **.** 来仅匹配子域。例如： **.y.com** 匹配 **x.y.com**，但不匹配 **y.com**。使用 ***** 绕过所有目的地的代理。您必须包含 vCenter 的 IP 地址以及用于其机器的 IP 范围。
4. 如果提供，安装程序会在 **openshift-config 命名空间中生成名为 user-ca- bundle** 的配置映射来保存额外的 CA 证书。如果您提供 **additionalTrustBundle** 和至少一个代理设置，则 **Proxy** 对象会被配置为引用 **trustedCA** 字段中的 **user-ca-bundle** 配置映射。然后，Cluster Network Operator 会创建一个 **trusted-ca-bundle** 配置映射，该配置映射将为 **trustedCA** 参数指定的内容与 RHCOS 信任捆绑包合并。 **additionalTrustBundle** 字段是必需的，除非代理的身份证书由来自 RHCOS 信任捆绑包的颁发机构签名。

注意

安装程序不支持代理的 **readinessEndpoints** 字段。

2. 保存该文件，并在安装 OpenShift Container Platform 时引用。

安装程序会创建一个名为 **cluster** 的集群范围代理，该代理使用提供的 **install-config.yaml** 文件中的代理设置。如果没有提供代理设置，仍然会创建一个 **cluster Proxy** 对象，但它会有一个空 **spec**。

注意

只支持名为 **cluster** 的 **Proxy** 对象，且无法创建额外的代理。

12.6.10. 指定高级网络配置

您可以通过为集群网络供应商指定额外的配置，使用高级配置自定义将集群整合到现有网络环境中。您只能在安装集群前指定高级网络配置。

重要

不支持修改安装程序创建的 OpenShift Container Platform 清单文件。支持应用您创建的清单文件，如以下流程所示。

先决条件

- 创建 **install-config.yaml** 文件并完成对其所做的任何修改。
- 为集群生成 Ignition 配置文件。

流程

1. 进入包含安装程序的目录并创建清单：

```
$ ./openshift-install create manifests --dir <installation_directory>
```

其中：

<installation_directory>
指定包含集群的 **install-config.yaml** 文件的目录名称。

2. 在 **<installation_directory>/manifests/** 目录下，为高级网络配置创建一个名为 **cluster-network-03-config.yml** 的 stub 清单文件：

```
$ cat <<EOF > <installation_directory>/manifests/cluster-network-03-config.yml
apiVersion: operator.openshift.io/v1
kind: Network
metadata:
  name: cluster
spec:
EOF
```

其中：

<installation_directory>

指定包含集群的 **manifests/** 目录的目录名称。

3. 在编辑器中打开 **cluster-network-03-config.yml** 文件，并为集群指定高级网络配置，如下例所示：

   为 OpenShift SDN 网络供应商指定不同的 VXLAN 端口

   ```
   apiVersion: operator.openshift.io/v1
   kind: Network
   metadata:
     name: cluster
   spec:
     defaultNetwork:
       openshiftSDNConfig:
         vxlanPort: 4800
   ```

4. 保存 **cluster-network-03-config.yml** 文件，再退出文本编辑器。

5. 可选：备份 **manifests/cluster-network-03-config.yml** 文件。创建集群时，安装程序会删除 **manifests/** 目录。

6. 删除定义 control plane 机器的 Kubernetes 清单文件以及计算 machineSets：

   ```
   $ rm -f openshift/99_openshift-cluster-api_master-machines-*.yaml openshift/99_openshift-cluster-api_worker-machineset-*.yaml
   ```

   由于您要自行创建和管理这些资源，因此不必初始化这些资源。

   - 您可以使用机器 API 来保留 MachineSet 文件来创建计算机器，但您必须更新对其的引用，以匹配您的环境。

12.6.11. Cluster Network Operator 配置

集群网络的配置作为 Cluster Network Operator (CNO) 配置的一部分被指定，并存储在名为 **cluster**的自定义资源（CR）对象中。CR 指定 **operator.openshift.io** API 组中的 **Network** API 的字段。

CNO 配置会在集群安装过程中从 **Network.config.openshift.io** API 组中的 **Network** API 继承以下字段，这些字段无法更改：

clusterNetwork

从中分配 pod IP 地址的 IP 地址池。

serviceNetwork

服务的 IP 地址池。

defaultNetwork.type

集群网络供应商，如 OpenShift SDN 或 OVN-Kubernetes。

您可以通过在名为 **cluster** 的 CNO 对象中设置 **defaultNetwork** 对象的字段来为集群指定集群网络供应商配置。

12.6.11.1. Cluster Network Operator 配置对象

Cluster Network Operator（CNO）的字段在下表中描述：

表 12.57. Cluster Network Operator 配置对象

字段	类型	描述
metadata.name	**字符串**	CNO 对象的名称。这个名称始终是 **cluster**。

spec.clusterNetwork	**数组**	用于指定从哪些 IP 地址块分配 Pod IP 地址以及分配给集群中每个节点的子网前缀长度的列表。例如：
		`spec:`
		`  clusterNetwork:`
		`  - cidr: 10.128.0.0/19`
		`  hostPrefix: 23`
		`  - cidr: 10.128.32.0/19`
		`  hostPrefix: 23`
		此值是只读的，并在 **install-config.yaml** 文件中指定。

spec.serviceNetwork	**数组**	服务的 IP 地址块。OpenShift SDN 和 OVN-Kubernetes Container Network Interface（CNI）网络供应商只支持服务网络具有单个 IP 地址块。例如：
		`spec:`
		`  serviceNetwork:`
		`  - 172.30.0.0/14`
		此值是只读的，并在 **install-config.yaml** 文件中指定。

spec.defaultNetwork	**对象**	为集群网络配置 Container Network Interface（CNI）集群网络供应商。
spec.kubeProxyConfig	**对象**	此对象的字段指定 kube-proxy 配置。如果您使用 OVN-Kubernetes 集群网络供应商，则 kube-proxy 的配置不会起作用。

defaultNetwork 对象配置
defaultNetwork 对象的值在下表中定义：

表 12.58. **defaultNetwork** 对象

字段	类型	描述

字段	类型	描述
type	**字符串**	**OpenShiftSDN** 或 **OVNKubernetes**。在安装过程中选择了集群网络供应商。集群安装后无法更改这个值。  **注意** OpenShift Container Platform 默认使用 OpenShift SDN Container Network Interface（CNI）集群网络供应商。
openshiftSDNConfig	**对象**	此对象仅对 OpenShift SDN 集群网络供应商有效。
ovnKubernetesConfig	**对象**	此对象仅对 OVN-Kubernetes 集群网络供应商有效。

配置 OpenShift SDN CNI 集群网络供应商
下表描述了 OpenShift SDN Container Network Interface（CNI）集群网络供应商的配置字段。

表 12.59. openshiftSDNConfig 对象

字段	类型	描述
mode	**字符串**	配置 OpenShift SDN 的网络隔离模式。默认值为 **NetworkPolicy**。 **Multitenant** 和 **Subnet** 的值可以向后兼容 OpenShift Container Platform 3.x，但不推荐这样做。集群安装后无法更改这个值。
mtu	**整数**	VXLAN 覆盖网络的最大传输单元 (MTU) 。这根据主网络接口的 MTU 自动探测。您通常不需要覆盖检测到的 MTU。 如果自动探测的值不是您期望的，请确认节点上主网络接口中的 MTU 是正确的。您不能使用这个选项更改节点上主网络接口的 MTU 值。 如果您的集群中的不同节点需要不同的 MTU 值，则必须将此值设置为比集群中的最低 MTU 值小 **50**。例如，如果集群中的某些节点的 MTU 为 **9001**，而某些节点的 MTU 为 **1500**，则必须将此值设置为 **1450**。 集群安装后无法更改这个值。

字段	类型	描述
vxlanPort	**整数**	用于所有 VXLAN 数据包的端口。默认值为 **4789**。集群安装后无法更改这个值。 如果您在虚拟环境中运行，并且现有节点是另一个 VXLAN 网络的一部分，那么可能需要更改此值。例如，当在 VMware NSX-T 上运行 OpenShift SDN 覆盖时，您必须为 VXLAN 选择一个备用端口，因为两个 SDN 都使用相同的默认 VXLAN 端口号。 在 Amazon Web Services (AWS) 上，您可以在端口 **9000** 和端口 **9999** 之间为 VXLAN 选择一个备用端口。

OpenShift SDN 配置示例

```
defaultNetwork:
  type: OpenShiftSDN
  openshiftSDNConfig:
    mode: NetworkPolicy
    mtu: 1450
    vxlanPort: 4789
```

配置 OVN-Kubernetes CNI 集群网络供应商

下表描述了 OVN-Kubernetes CNI 集群网络供应商的配置字段。

表 12.60. ovnKubernetesConfig 对象

字段	类型	描述
mtu	**整数**	Geneve（Generic Network Virtualization Encapsulation）覆盖网络的最大传输单元（MTU）。这根据主网络接口的 MTU 自动探测。您通常不需要覆盖检测到的 MTU。 如果自动探测的值不是您期望的，请确认节点上主网络接口中的 MTU 是正确的。您不能使用这个选项更改节点上主网络接口的 MTU 值。 如果您的集群中的不同节点需要不同的 MTU 值，则必须将此值设置为比集群中的最低 MTU 值小 **100**。例如，如果集群中的某些节点的 MTU 为 **9001**，而某些节点的 MTU 为 **1500**，则必须将此值设置为 **1400**。 集群安装后无法更改这个值。
genevePort	**整数**	用于所有 Geneve 数据包的端口。默认值为 **6081**。集群安装后无法更改这个值。

OVN-Kubernetes 配置示例

```
defaultNetwork:
  type: OVNKubernetes
  ovnKubernetesConfig:
```

```
    mtu: 1400
    genevePort: 6081
```

kubeProxyConfig 对象配置
kubeProxyConfig 对象的值在下表中定义：

表 12.61. kubeProxyConfig 对象

字段	类型	描述
iptablesSyncPeriod	**字符串**	**iptables** 规则的刷新周期。默认值为 **30s**。有效的后缀包括 **s**、**m** 和 **h**，具体参见 Go **time** 软件包文档。 **注意** 由于 OpenShift Container Platform 4.3 及更高版本中引进了性能上的改进，现在不再需要调整 **iptablesSyncPeriod** 参数。

proxyArguments.iptables-min-sync-period	**数组**	刷新 **iptables** 规则前的最短时长。此字段确保刷新的频率不会过于频繁。有效的后缀包括 **s**、**m** 和 **h**，具体参见 Go **time** 软件包。默认值为：
		
		`kubeProxyConfig:`
		`  proxyArguments:`
		`  iptables-min-sync-period:`
		`  - 0s`

12.6.12. 创建 Ignition 配置文件

由于需要手工启动集群机器，因此您必须生成 Ignition 配置文件，集群需要它来创建其机器。

重要

- 安装程序生成的 Ignition 配置文件包含在 24 小时后过期的证书，然后在过期时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外情况是，您需要手动批准待处理的 **node-bootstrapper** 证书签名请求（CSR）来恢复 kubelet 证书。如需更多信息，请参阅 *从过期的 control plane 证书中恢复* 的文档。
- 建议您在生成 12 小时后使用 Ignition 配置文件，因为集群安装后 24 小时证书从 16 小时轮转至 22 小时。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中运行证书更新时避免安装失败。

先决条件

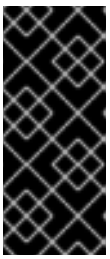
- 获取 OpenShift Container Platform 安装程序以及集群的 pull secret。对于受限网络安装，这些文件位于您的堡垒主机上。

流程

- 获取 Ignition 配置文件：

```
$ ./openshift-install create ignition-configs --dir <installation_directory> 1
```

1 对于 **<installation_directory>**，请指定用于保存安装程序所创建的文件的目录名称。



重要

如果您创建了 **install-config.yaml** 文件，请指定包含该文件的目录。否则，指定一个空目录。一些安装信息，如 bootstrap X.509 证书，有较短的过期间隔，因此不要重复使用安装目录。如果要重复使用另一个集群安装中的个别文件，可以将其复制到您的目录中。但是，一些安装数据的文件名可能会在发行版本之间有所改变。从 OpenShift Container Platform 老版本中复制安装文件时要格外小心。

该目录中将生成以下文件：

```
.
├── auth
│   ├── kubeadmin-password
│   └── kubeconfig
├── bootstrap.ign
├── master.ign
├── metadata.json
└── worker.ign
```

12.6.13. 提取基础架构名称

Ignition 配置文件包含一个唯一的集群标识符,您可以使用它在 VMware Cloud 中唯一地标识您的集群。如果计划使用集群标识符作为虚拟机文件夹的名称,您必须提取它。

先决条件

- 获取 OpenShift Container Platform 安装程序以及集群的 pull secret。
- 已为集群生成 Ignition 配置文件。
- 安装了 **jq** 软件包。

流程

- 要从 Ignition 配置文件元数据中提取和查看基础架构名称，请运行以下命令：

```
$ jq -r .infraID <installation_directory>/metadata.json 1
```

1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

输出示例

```
openshift-vw9j6 1
```

1 此命令的输出是您的集群名称和随机字符串。

12.6.14. 在 vSphere 中创建 Red Hat Enterprise Linux CoreOS (RHCOS) 机器

在 VMware vSphere 上安装包含用户置备基础架构的集群前，您必须在 vSphere 主机上创建 RHCOS 机器供其使用。

先决条件

- 已获取集群的 Ignition 配置文件。
- 您可以从计算机访问 HTTP 服务器，并且您创建的机器也可以访问该服务器。
- 您已创建了 vSphere 集群。

流程

1. 将名为 **<installation_directory>/bootstrap.ign** 的 bootstrap Ignition 配置文件上传到 HTTP 服务器，该配置文件是由安装程序创建的。记下此文件的 URL。
2. 将 bootstrap 节点的以下辅助 Ignition 配置文件保存到计算机中，存为 **<installation_directory>/merge-bootstrap.ign** ：

```
{
  "ignition": {
    "config": {
      "merge": [
        {
          "source": "<bootstrap_ignition_config_url>", 1
          "verification": {}
        }
      ]
    },
    "timeouts": {},
    "version": "3.1.0"
  },
  "networkd": {},
  "passwd": {},
  "storage": {},
  "systemd": {}
}
```

1 指定您托管的 bootstrap Ignition 配置文件的 URL。

为 bootstrap 机器创建虚拟机 (VM) 时，您要使用此 Ignition 配置文件。

3. 找到安装程序创建的以下 Ignition 配置文件：
   - **<installation_directory>/master.ign**
   - **<installation_directory>/worker.ign**
   - **<installation_directory>/merge-bootstrap.ign**

4. 将 Ignition 配置文件转换为 Base64 编码。在此流程中，您必须将这些文件添加到虚拟机中的额外配置参数 **guestinfo.ignition.config.data** 中。
例如，如果您使用 Linux 操作系统，可以使用 **base64** 命令来编码这些文件。

```
$ base64 -w0 <installation_directory>/master.ign > <installation_directory>/master.64
```

```
$ base64 -w0 <installation_directory>/worker.ign > <installation_directory>/worker.64
```

```
$ base64 -w0 <installation_directory>/merge-bootstrap.ign > <installation_directory>/merge-bootstrap.64
```

重要

如果您计划在安装完成后在集群中添加更多计算机器，请不要删除这些文件。

5. 获取 RHCOS OVA 镜像。镜像位于 RHCOS 镜像镜像页面。

重要

RHCOS 镜像可能不会随着 OpenShift Container Platform 的每一发行版本都有改变。您必须下载一个最高版本的镜像，其版本号应小于或等于您安装的 OpenShift Container Platform 版本。如果可用，请使用与 OpenShift Container Platform 版本匹配的镜像版本。

文件名包含 OpenShift Container Platform 版本号，格式为 **rhcos-vmware.<architecture>.ova**。

6. 在 vSphere 客户端中，在数据中心中创建一个文件夹来存储您的虚拟机。
   a. 点击 **VMs and Templates** 视图。
   b. 右键点击您的数据中心名称。
   c. 点击 **New Folder → New VM and Template Folder**。
   d. 在显示的窗口中输入文件夹名称。如果您没有在 **install-config.yaml** 文件中指定现有文件夹，请创建一个文件夹，其名称与基础架构 ID 相同。您可以使用这个文件夹名称，因此 vCenter 会在适当的位置为 Workspace 配置动态置备存储。

7. 在 vSphere 客户端中，为 OVA 镜像创建一个模板，然后根据需要克隆模板。

注意

在以下步骤中，您将创建一个模板，然后克隆所有集群机器的模板。然后，在置备虚拟机时，为该克隆的机器类型提供 Ignition 配置文件的位置。

   a. 在 **Hosts and Clusters** 选项卡中，右键点击您的集群名称并选择 **Deploy OVF Template**。
   b. 在 **Select an OVF** 选项卡中，指定您下载的 RHCOS OVA 文件的名称。
   c. 在 **Select a name and folder** 选项卡中，为您的模板设置 **虚拟机名称**，如 **Template-RHCOS**。点击 vSphere 集群的名称并选择您在上一步中创建的文件夹。

d. 在 **Select a compute resource** 选项卡中，点击您的 vSphere 集群名称。

e. 在 **Select storage** 选项卡中，配置虚拟机的存储选项。

- 根据您的存储要求，选择**Thin Provision** 或 **Thick Provision**。
- 选择您在 **install-config.yaml** 文件中指定的数据存储。

f. 在 **Select network** 选项卡中，指定您为集群配置的网络（如果可用）。

g. 在创建 OVF 模板时，请不要在 **Customize template** 选项卡上指定值，或者不要再配置模板。

重要

不要启动原始虚拟机模板。VM 模板必须保持关闭状态，必须为新的 RHCOS 机器克隆。启动虚拟机模板会将虚拟机模板配置为平台上的虚拟机，这样可防止它被用作计算机集可以应用配置的模板。

8. 部署模板后，为集群中的机器部署虚拟机。

a. 右键点击模板的名称，再点击 **Clone → Clone to Virtual Machine**。

b. 在 **Select a name and folder** 选项卡中，指定虚拟机的名称。名称中可以包括机器类型，如 **control-plane-0** 或 **compute-1**。

c. 在 **Select a name and folder** 选项卡中，选择您为集群创建的文件夹名称。

d. 在 **Select a compute resource** 选项卡中，选择数据中心中的主机名称。
对于 bootstrap 机器，指定您托管的 bootstrap Ignition 配置文件的 URL。

e. 可选：在 **Select storage** 选项卡中，自定义存储选项。

f. 在 **Select clone options** 中，选择 **Customize this virtual machine's hardware**。

g. 在 **Customize hardware** 选项卡中，点击 **VM Options → Advanced**。

- 可选：覆盖 vSphere 中的默认 DHCP 网络。启用静态 IP 网络：

i. 设置静态 IP 配置：

```
$ export IPCFG="ip=<ip>::<gateway>:<netmask>:<hostname>:<iface>:none
nameserver=srv1 [nameserver=srv2 [nameserver=srv3 [...]]]"
```

示例命令

```
$ export IPCFG="ip=192.168.100.101::192.168.100.254:255.255.255.0:::none
nameserver=8.8.8.8"
```

ii. 在从 vSphere 中的 OVA 引导虚拟机前，设置 **guestinfo.afterburn.initrd.network-kargs** 属性：

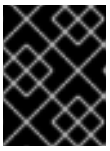
```
$ govc vm.change -vm "<vm_name>" -e "guestinfo.afterburn.initrd.network-
kargs=${IPCFG}"
```

- 可选：在出现集群性能问题时，从 **Latency Sensitivity** 列表中选择 **High**。确定虚拟机的 CPU 和内存保留有以下值：
  - 内存保留值必须等于其配置的内存大小。
  - CPU 保留值必须至少是低延迟虚拟 CPU 的数量，乘以测量的物理 CPU 速度。
- 点击 **Edit Configuration**，然后在 **Configuration Parameters** 窗口中点击 **Add Configuration Params**。定义以下参数名称和值：
  - **guestinfo.ignition.config.data** ：找到您在此流程中创建的 base-64 编码文件，并粘贴此机器类型的 base64 编码 Ignition 配置文件的内容。
  - **guestinfo.ignition.config.data.encoding**：指定 **base64**。
  - **disk.EnableUUID**：指定 **TRUE**。

h. 在 **Customize hardware** 选项卡的 **Virtual Hardware** 面板中，根据需要修改指定的值。确保 RAM、CPU 和磁盘存储的数量满足机器类型的最低要求。

i. 完成配置并打开虚拟机电源。

9. 对于每台机器，按照前面的步骤为集群创建其余的机器。



重要

此刻您必须创建 bootstrap 和 control plane 机器。由于计算机器中已默认部署了一些 Pod，因此在安装集群前，还要创建至少两台计算机器。

12.6.15. 在 vSphere 中创建更多 Red Hat Enterprise Linux CoreOS（RHCOS）机器

您可以为集群创建更多计算机器，在 VMware vSphere 上使用用户置备的基础架构。

先决条件

- 获取计算机器的 Base64 编码 Ignition 文件。
- 您可以访问您为集群创建的 vSphere 模板。

流程

1. 部署模板后，为集群中的机器部署虚拟机。

a. 右键点击模板的名称，再点击 **Clone → Clone to Virtual Machine**。

b. 在 **Select a name and folder** 选项卡中，指定虚拟机的名称。您可以在名称中包含机器类型，如 **compute-1**。

c. 在 **Select a name and folder** 选项卡中，选择您为集群创建的文件夹名称。

d. 在 **Select a compute resource** 选项卡中，选择数据中心中的主机名称。

e. 可选：在 **Select storage** 选项卡中，自定义存储选项。

f. 在 **Select clone options** 中，选择 **Customize this virtual machine's hardware**。

g. 在 **Customize hardware** 选项卡中，点击 **VM Options → Advanced**。

- 从 **Latency Sensitivity** 列表中选择 **High**。
- 点击 **Edit Configuration**，然后在 **Configuration Parameters** 窗口中点击 **Add Configuration Params**。定义以下参数名称和值：
  - **guestinfo.ignition.config.data**：粘贴此机器类型的 Base64 编码计算 Ignition 配置文件的内容。
  - **guestinfo.ignition.config.data.encoding**：指定 **base64**。
  - **disk.EnableUUID**：指定 **TRUE**。

h. 在 **Customize hardware** 选项卡的 **Virtual Hardware** 面板中，根据需要修改指定的值。确保 RAM、CPU 和磁盘存储的数量满足机器类型的最低要求。另外，如果有多个可用的网络，请确定在 **Add network adapter 中** 选择正确的网络。

i. 完成配置并打开虚拟机电源。

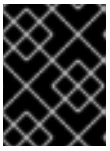
2. 继续为集群创建更多计算机器。

12.6.16. 磁盘分区

在大多数情况下，数据分区最初是由安装 RHCOS 而不是安装另一个操作系统来创建的。在这种情况下，OpenShift Container Platform 安装程序应该被允许配置磁盘分区。

但是，在安装 OpenShift Container Platform 节点时，在两种情况下您可能需要覆盖默认分区：

- 创建单独的分区：对于在空磁盘中的 greenfield 安装，您可能想要在分区中添加单独的存储。这只在生成 **/var** 或者一个 **/var** 独立分区的子目录（如 **/var/lib/etcd** ）时被正式支持，但不支持两者。



重要

Kubernetes 只支持两个文件系统分区。如果您在原始配置中添加多个分区，Kubernetes 无法监控所有这些分区。

- 保留现有分区：对于 brownfield 安装，您要在现有节点上重新安装 OpenShift Container Platform，并希望保留从之前的操作系统中安装的数据分区，对于 **coreos-installer** 来说，引导选项和选项都允许您保留现有数据分区。

创建一个独立的 /var 分区

通常情况下，OpenShift Container Platform 的磁盘分区应该留给安装程序。然而，在有些情况下您可能需要在文件系统的一部分中创建独立分区。

OpenShift Container Platform 支持添加单个分区来将存储附加到 **/var** 分区或 **/var** 的子目录。例如：

- **/var/lib/containers**：保存镜像相关的内容，随着更多镜像和容器添加到系统中，它所占用的存储会增加。
- **/var/lib/etcd**：保存您可能希望保持独立的数据，比如 etcd 存储的性能优化。
- **/var**：保存您希望独立保留的数据，用于特定目的（如审计）。

单独存储 **/var** 目录的内容可方便地根据需要对区域扩展存储，并可以在以后重新安装 OpenShift Container Platform 时保持该数据地完整。使用这个方法，您不必再次拉取所有容器，在更新系统时也无法复制大量日志文件。

因为 **/var** 在进行一个全新的 Red Hat Enterprise Linux CoreOS（RHCOS）安装前必需存在，所以这个流程会在 OpenShift Container Platform 安装过程的 **openshift-install** 准备阶段插入的机器配置来设置独立的 **/var** 分区。

流程

1. 创建存放 OpenShift Container Platform 安装文件的目录：

```
$ mkdir $HOME/clusterconfig
```

2. 运行 **openshift-install** 在 **manifest** 和 **openshift** 子目录中创建一组文件。在出现提示时回答系统问题：

```
$ openshift-install create manifests --dir $HOME/clusterconfig
? SSH Public Key ...
$ ls $HOME/clusterconfig/openshift/
99_kubeadmin-password-secret.yaml
99_openshift-cluster-api_master-machines-0.yaml
99_openshift-cluster-api_master-machines-1.yaml
99_openshift-cluster-api_master-machines-2.yaml
...
```

3. 创建 **MachineConfig** 对象并将其添加到 **openshift** 目录中的一个文件中。例如，把文件命名为 **98-var-partition.yaml**，将磁盘设备名称改为 **worker** 系统中存储设备的名称，并根据情况设置存储大小。这个示例将 **/var** 目录放在一个单独的分区中：

```
apiVersion: machineconfiguration.openshift.io/v1
kind: MachineConfig
metadata:
  labels:
    machineconfiguration.openshift.io/role: worker
  name: 98-var-partition
spec:
  config:
    ignition:
      version: 3.1.0
    storage:
      disks:
      - device: /dev/<device_name> 1
        partitions:
        - label: var
          startMiB: <partition_start_offset> 2
          sizeMiB: <partition_size> 3
      filesystems:
        - device: /dev/disk/by-partlabel/var
          path: /var
          format: xfs
    systemd:
      units:
        - name: var.mount 4
          enabled: true
          contents: |
            [Unit]
            Before=local-fs.target
```

```
    [Mount]
    What=/dev/disk/by-partlabel/var
    Where=/var
    Options=defaults,prjquota 5
    [Install]
    WantedBy=local-fs.target
```

1. 要分区的磁盘的存储设备名称。
2. 当在引导磁盘中添加数据分区时，推荐最少使用 25000MB。root 文件系统会自动重新定义大小使其占据所有可用空间（最多到指定的偏移值）。如果没有指定值，或者指定的值小于推荐的最小值，则生成的 root 文件系统会太小，而在以后进行的 RHCOS 重新安装可能会覆盖数据分区的开始部分。
3. 数据分区的大小（以兆字节为单位）。
4. 挂载单元的名称必须与 **Where=** 指令中指定的目录匹配。例如，对于挂载在 **/var/lib/containers** 上的文件系统，该单元必须命名为 **var-lib-containers.mount**。
5. 对于用于容器存储的文件系统，必须启用 **prjquota** 挂载选项。

注意

在创建独立 **/var** 分区时，如果不同的实例类型没有相同的设备名称，则无法将不同的实例类型用于 worker 节点。

4. 再次运行 **openshift-install**，从 **manifest** 和 **openshift** 子目录中的一组文件创建 Ignition 配置：

```
$ openshift-install create ignition-configs --dir $HOME/clusterconfig
$ ls $HOME/clusterconfig/
auth  bootstrap.ign  master.ign  metadata.json  worker.ign
```

现在，您可以使用 Ignition 配置文件作为 vSphere 安装程序的输入来安装 Red Hat Enterprise Linux CoreOS（RHCOS）系统。

12.6.17. 创建集群

要创建 OpenShift Container Platform 集群，请等待您通过安装程序生成的 Ignition 配置文件所置备的机器上完成 bootstrap 过程。

先决条件

- 为集群创建所需的基础架构。
- 已获得安装程序并为集群生成了 Ignition 配置文件。
- 已使用 Ignition 配置文件为集群创建 RHCOS 机器。
- 您的机器可直接访问互联网，或者可以使用 HTTP 或 HTTPS 代理。

流程

1. 监控 bootstrap 过程：

```
$ ./openshift-install --dir <installation_directory> wait-for bootstrap-complete \ 1
    --log-level=info 2
```

1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

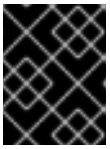
2 要查看不同的安装详情，请指定 **warn**、**debug** 或 **error**，而不要指定 **info**。

输出示例

```
INFO Waiting up to 30m0s for the Kubernetes API at https://api.test.example.com:6443...
INFO API v1.19.0 up
INFO Waiting up to 30m0s for bootstrapping to complete...
INFO It is now safe to remove the bootstrap resources
```

Kubernetes API 服务器提示已在 control plane 机器上完成 bootstrap 时，命令运行成功。

2. bootstrap 过程完成后，请从负载均衡器中删除 bootstrap 机器。



重要

此时您必须从负载均衡器中删除 bootstrap 机器。您还可以删除或重新格式化机器本身。

12.6.18. 使用 CLI 登录到集群

您可以通过导出集群 **kubeconfig** 文件，以默认系统用户身份登录集群。**kubeconfig** 文件包含关于集群的信息，供 CLI 用于将客户端连接到正确集群和 API 服务器。该文件特只适用于一个特定的集群，在 OpenShift Container Platform 安装过程中创建。

先决条件

- 已部署了 OpenShift Container Platform 集群。
- 已安装 **oc** CLI。

流程

1. 导出 **kubeadmin** 凭证：

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

2. 使用导出的配置，验证能否成功运行 **oc** 命令：

```
$ oc whoami
```

输出示例

```
system:admin
```

12.6.19. 批准机器的证书签名请求

将机器添加到集群时，会为您添加的每台机器生成两个待处理证书签名请求（CSR）。您必须确认这些 CSR 已获得批准，或根据需要自行批准。客户端请求必须首先被批准，然后是服务器请求。

先决条件

- 您已将机器添加到集群中。

流程

1. 确认集群可以识别这些机器：

```
$ oc get nodes
```

输出示例

```
NAME     STATUS   ROLES   AGE  VERSION
master-0 Ready    master  63m  v1.19.0
master-1 Ready    master  63m  v1.19.0
master-2 Ready    master  64m  v1.19.0
```

输出将列出您创建的所有机器。

注意

在一些 CSR 被批准前，以上输出可能不包括计算节点（也称为 worker 节点）。

2. 检查待处理的 CSR，并确保可以看到添加到集群中的每台机器都有 **Pending** 或 **Approved** 状态的客户端请求：

```
$ oc get csr
```

输出示例

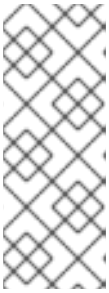
```
NAME        AGE    REQUESTOR                                                   CONDITION
csr-8b2br   15m    system:serviceaccount:openshift-machine-config-operator:node-
bootstrapper   Pending
csr-8vnps   15m    system:serviceaccount:openshift-machine-config-operator:node-
bootstrapper   Pending
...
```

在本例中，两台机器加入了集群。您可能会在列表中看到更多已批准的 CSR。

3. 如果 CSR 没有获得批准，请在所添加机器的所有待处理 CSR 都处于 **Pending** 状态后，为您的集群机器批准这些 CSR：

注意

由于 CSR 会自动轮转，因此请在将机器添加到集群后一小时内批准您的 CSR。如果没有在一小时内批准，证书将会轮转，每个节点将会存在多个证书。您必须批准所有这些证书。批准客户端 CSR 后， Kubelet 为服务证书创建辅助 CSR，这需要手动批准。然后，如果 Kubelet 请求具有相同参数的新证书，则 **machine-approver** 会自动批准后续服务证书续订请求。



注意

对于在未启用机器 API 的平台中运行的集群，如裸机和其他用户置备的基础架构，必须采用一种方法自动批准 kubelet 提供证书请求（CSR）。如果没有批准请求，则 **oc exec**、**oc rsh** 和 **oc logs** 命令将无法成功，因为 API 服务器连接到 kubelet 时需要服务证书。与 Kubelet 端点联系的任何操作都需要此证书批准。这个方法必须监视新的 CSR，确认 CSR 由 **system:node** 或 **system:admin** 组中的 **node-bootstrapper** 服务帐户提交，并确认节点的身份。

- 若要单独批准，请对每个有效的 CSR 运行以下命令：

```
$ oc adm certificate approve <csr_name> 1
```

1 **<csr_name>** 是当前 CSR 列表中 CSR 的名称。

- 要批准所有待处理的 CSR，请运行以下命令：

```
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}{{"\n"}}
{{end}}{{end}}' | xargs --no-run-if-empty oc adm certificate approve
```

注意

在有些 CSR 被批准前，一些 Operator 可能无法使用。

4. 现在，您的客户端请求已被批准，您必须查看添加到集群中的每台机器的服务器请求：

```
$ oc get csr
```

输出示例

```
NAME        AGE    REQUESTOR                                                   CONDITION
csr-bfd72   5m26s  system:node:ip-10-0-50-126.us-east-2.compute.internal
Pending
csr-c57lv   5m26s  system:node:ip-10-0-95-157.us-east-2.compute.internal
Pending
...
```

5. 如果剩余的 CSR 没有被批准，且处于 **Pending** 状态，请批准集群机器的 CSR：

- 若要单独批准，请对每个有效的 CSR 运行以下命令：

```
$ oc adm certificate approve <csr_name> 1
```

1 **<csr_name>** 是当前 CSR 列表中 CSR 的名称。

- 要批准所有待处理的 CSR，请运行以下命令：

```
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}{{"\n"}}
{{end}}{{end}}' | xargs oc adm certificate approve
```

6. 批准所有客户端和服务器 CSR 后，器将处于 **Ready** 状态。运行以下命令验证：

```
$ oc get nodes
```

输出示例

```
NAME      STATUS    ROLES   AGE  VERSION
master-0  Ready     master  73m  v1.20.0
master-1  Ready     master  73m  v1.20.0
master-2  Ready     master  74m  v1.20.0
worker-0  Ready     worker  11m  v1.20.0
worker-1  Ready     worker  11m  v1.20.0
```

注意

批准服务器 CSR 后可能需要几分钟时间让机器转换为 **Ready** 状态。

其他信息

- 如需有关 CSR 的更多信息，请参阅证书签名请求。

12.6.20. 初始 Operator 配置

在 control plane 初始化后，您必须立即配置一些 Operator 以便它们都可用。

先决条件

- 您的 control plane 已初始化。

流程

1. 观察集群组件上线：

```
$ watch -n5 oc get clusteroperators
```

输出示例

```
NAME                                       VERSION AVAILABLE   PROGRESSING   DEGRADED
SINCE
authentication                              4.6.0   True        False         False      3h56m
cloud-credential                            4.6.0   True        False         False      29h
cluster-autoscaler                          4.6.0   True        False         False      29h
config-operator                             4.6.0   True        False         False      6h39m
console                                    4.6.0   True        False         False      3h59m
csi-snapshot-controller                          4.6.0   True        False         False      4h12m
```

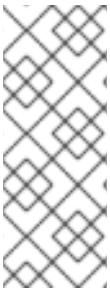
```
dns                                        4.6.0   True        False         False      4h15m
etcd                                       4.6.0   True        False         False      29h
image-registry                             4.6.0   True        False         False      3h59m
ingress                                    4.6.0   True        False         False      4h30m
insights                                   4.6.0   True        False         False      29h
kube-apiserver                             4.6.0   True        False         False      29h
kube-controller-manager                    4.6.0   True        False         False      29h
kube-scheduler                             4.6.0   True        False         False      29h
kube-storage-version-migrator              4.6.0   True        False         False      4h2m
machine-api                                4.6.0   True        False         False      29h
machine-approver                           4.6.0   True        False         False      6h34m
machine-config                             4.6.0   True        False         False      3h56m
marketplace                                4.6.0   True        False         False      4h2m
monitoring                                 4.6.0   True        False         False      6h31m
network                                    4.6.0   True        False         False      29h
node-tuning                                4.6.0   True        False         False      4h30m
openshift-apiserver                        4.6.0   True        False         False      3h56m
openshift-controller-manager               4.6.0   True        False         False      4h36m
openshift-samples                          4.6.0   True        False         False      4h30m
operator-lifecycle-manager                 4.6.0   True        False         False      29h
operator-lifecycle-manager-catalog         4.6.0   True        False         False      29h
operator-lifecycle-manager-packageserver   4.6.0   True        False         False      3h59m
service-ca                                 4.6.0   True        False         False      29h
storage                                    4.6.0   True        False         False      4h30m
```

2. 配置不可用的 Operator。

12.6.20.1. 安装过程中删除的镜像 registry

在不提供可共享对象存储的平台上，OpenShift Image Registry Operator bootstraps 本身的状态是 **Removed**。这允许 **openshift-installer** 在这些平台类型上完成安装。

将 **ManagementState** Image Registry Operator 配置从 **Removed** 改为 **Managed**。



注意

Prometheus 控制台提供了一个 **ImageRegistryRemoved** 警报，例如：

"Image Registry has been removed.**ImageStreamTags**, **BuildConfigs** and **DeploymentConfigs** which reference **ImageStreamTags** may not work as expected.Please configure storage and update the config to **Managed** state by editing configs.imageregistry.operator.openshift.io."

12.6.20.2. 镜像 registry 存储配置

对于不提供默认存储的平台，Image Registry Operator 最初将不可用。安装后，您必须配置 registry 使用的存储，这样 Registry Operator 才可用。

示配置生产集群所需的持久性卷的说明。如果适用，显示有关将空目录配置为存储位置的说明，该位置只可用于非生产集群。

另外还提供了在升级过程中使用 **Recreate** rollout 策略来允许镜像 registry 使用块存储类型的说明。

12.6.20.2.1. 为 VMware vSphere 配置块 registry 存储

在作为集群管理员升级时，要允许镜像 registry 使用块存储类型，如 vSphere Virtual Machine Disk（VMDK），您可以使用 **Recreate** rollout 策略。

重要

支持块存储卷，但不建议将其用于生产环境中的镜像 registry。在块存储上配置 registry 的安装不具有高可用性，因为 registry 无法拥有多个副本。

流程

1. 要将镜像 registry 存储设置为块存储类型，对 registry 进行补丁，使其使用 **Recreate** rollout 策略，且仅使用 **1** 个副本运行：

```
$ oc patch config.imageregistry.operator.openshift.io/cluster --type=merge -p '{"spec":
{"rolloutStrategy":"Recreate","replicas":1}}'
```

2. 为块存储设备置备 PV，并为该卷创建 PVC。请求的块卷使用 ReadWriteOnce（RWO）访问模式。

   a. 创建包含以下内容的 **pvc.yaml** 文件以定义 VMware vSphere **PersistentVolumeClaim**：

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: image-registry-storage 1
  namespace: openshift-image-registry 2
spec:
  accessModes:
  - ReadWriteOnce 3
  resources:
    requests:
      storage: 100Gi 4
```

   1. 代表 **PersistentVolumeClaim** 对象的唯一名称。
   2. **PersistentVolumeClaim** 对象的命名空间，即 **openshift-image-registry**。
   3. 持久性卷声明的访问模式。使用 **ReadWriteOnce** 时，单个节点可以通过读写权限挂载这个卷。
   4. 持久性卷声明的大小。

   b. 从文件创建 **PersistentVolumeClaim** 对象：

```
$ oc create -f pvc.yaml -n openshift-image-registry
```

3. 编辑 registry 配置，使其可以正确引用 PVC:

```
$ oc edit config.imageregistry.operator.openshift.io -o yaml
```

输出示例

```
storage:
  pvc:
    claim: 1
```

1 通过创建自定义 PVC，您可以将 **claim** 字段留空以用于默认自动创建 **image-registry-storage** PVC。

有关配置 registry 存储以便引用正确的 PVC 的说明，请参阅为 vSphere 配置 registry。

12.6.21. 在用户置备的基础架构上完成安装

完成 Operator 配置后，可以在您提供的基础架构上完成集群安装。

先决条件

- 您的 control plane 已初始化。
- 已完成初始 Operator 配置。

流程

1. 使用以下命令确认所有集群组件都已在线：

```
$ watch -n5 oc get clusteroperators
```

输出示例

```
NAME                                       VERSION AVAILABLE   PROGRESSING   DEGRADED
SINCE
authentication                             4.6.0   True        False         False     3h56m
cloud-credential                           4.6.0   True        False         False     29h
cluster-autoscaler                         4.6.0   True        False         False     29h
config-operator                            4.6.0   True        False         False     6h39m
console                                    4.6.0   True        False         False     3h59m
csi-snapshot-controller                    4.6.0   True        False         False     4h12m
dns                                        4.6.0   True        False         False     4h15m
etcd                                       4.6.0   True        False         False     29h
image-registry                             4.6.0   True        False         False     3h59m
ingress                                    4.6.0   True        False         False     4h30m
insights                                   4.6.0   True        False         False     29h
kube-apiserver                             4.6.0   True        False         False     29h
kube-controller-manager                    4.6.0   True        False         False     29h
kube-scheduler                             4.6.0   True        False         False     29h
kube-storage-version-migrator              4.6.0   True        False         False     4h2m
machine-api                                4.6.0   True        False         False     29h
machine-approver                           4.6.0   True        False         False     6h34m
machine-config                             4.6.0   True        False         False     3h56m
marketplace                                4.6.0   True        False         False     4h2m
monitoring                                 4.6.0   True        False         False     6h31m
network                                    4.6.0   True        False         False     29h
node-tuning                                4.6.0   True        False         False     4h30m
openshift-apiserver                        4.6.0   True        False         False     3h56m
openshift-controller-manager               4.6.0   True        False         False     4h36m
```

```
openshift-samples                          4.6.0   True        False      False      4h30m
operator-lifecycle-manager                 4.6.0   True        False      False      29h
operator-lifecycle-manager-catalog         4.6.0   True        False      False      29h
operator-lifecycle-manager-packageserver   4.6.0   True        False      False      3h59m
service-ca                                 4.6.0   True        False      False      29h
storage                                    4.6.0   True        False      False      4h30m
```

或者，通过以下命令，如果所有集群都可用您会接到通知。它还检索并显示凭证：

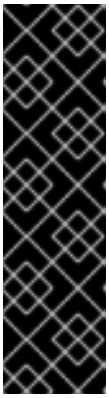
```
$ ./openshift-install --dir <installation_directory> wait-for install-complete 1
```

1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

输出示例

```
INFO Waiting up to 30m0s for the cluster to initialize...
```

Cluster Version Operator 完成从 Kubernetes API 服务器部署 OpenShift Container Platform 集群时，命令运行成功。



重要

- 安装程序生成的 Ignition 配置文件包含在 24 小时后过期的证书，然后在过期时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外情况是，您需要手动批准待处理的 **node-bootstrapper** 证书签名请求（CSR）来恢复 kubelet 证书。如需更多信息，请参阅*从过期的 control plane 证书中恢复*的文档。
- 建议您在生成 12 小时后使用 Ignition 配置文件，因为集群安装后 24 小时证书从 16 小时轮转至 22 小时。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中运行证书更新时避免安装失败。

2. 确认 Kubernetes API 服务器正在与 pod 通信。

   a. 要查看所有 pod 的列表，请使用以下命令：

   ```
   $ oc get pods --all-namespaces
   ```

   输出示例

   ```
   NAMESPACE                       NAME                                         READY   STATUS
   RESTARTS   AGE
   openshift-apiserver-operator    openshift-apiserver-operator-85cb746d55-zqhs8   1/1
   Running    1          9m
   openshift-apiserver             apiserver-67b9g                               1/1     Running     0
   3m
   openshift-apiserver             apiserver-ljcmx                               1/1     Running     0
   1m
   openshift-apiserver             apiserver-z25h4                               1/1     Running     0
   2m
   openshift-authentication-operator authentication-operator-69d5d8bf84-vh2n8       1/1
   Running    0          5m
   ...
   ```

b. 使用以下命令，查看上一命令的输出中所列 pod 的日志：

```
$ oc logs <pod_name> -n <namespace> 1
```

1 指定 pod 名称和命名空间，如上一命令的输出中所示。

如果 pod 日志显示，Kubernetes API 服务器可以与集群机器通信。

您可以按照将计算机器添加到 vSphere 的内容，在集群安装完成后添加额外的计算机器。

12.6.22. 备份 VMware vSphere 卷

OpenShift Container Platform 将新卷作为独立持久性磁盘置备，以便在集群中的任何节点上自由附加和分离卷。因此，无法备份使用快照的卷，也无法从快照中恢复卷。如需更多信息，请参阅 快照限制。

流程

要创建持久性卷的备份：

1. 停止使用持久性卷的应用程序。
2. 克隆持久性卷。
3. 重启应用程序。
4. 创建克隆的卷的备份。
5. 删除克隆的卷。

12.6.23. OpenShift Container Platform 的 Telemetry 访问

在 OpenShift Container Platform 4.6 中，默认运行的 Telemetry 服务提供有关集群健康状况和成功更新的指标，需要访问互联网。如果您的集群连接到互联网，Telemetry 会自动运行，而且集群会注册到 OpenShift Cluster Manager。

确认 OpenShift Cluster Manager 清单正确后，可以由 Telemetry 自动维护，也可以使用 OpenShift Cluster Manager 手动维护，使用订阅监控来跟踪帐户或多集群级别的 OpenShift Container Platform 订阅。

其他资源

- 有关 Telemetry 服务的更多信息，请参阅关于远程健康监控。

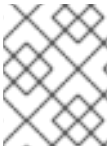
12.6.24. 后续步骤

- 自定义集群。
- 如果需要，您可以选择不使用远程健康报告。
- 设置 registry 并配置 registry 存储。

12.7. 在带有用户置备的受限网络中的 VMC 上安装集群

在 OpenShift Container Platform 版本 4.6 中，您可以通过将其部署到 VMware Cloud（VMC）on AWS 来在受限网络的 VMware vSphere 基础架构上安装集群。

为 OpenShift Container Platform 部署配置了 VMC 环境后，您要使用堡垒管理主机中的 OpenShift Container Platform 安装程序，并位于 VMC 环境中。安装程序和 control plane 可以自动部署和管理 OpenShift Container Platform 集群所需的资源。

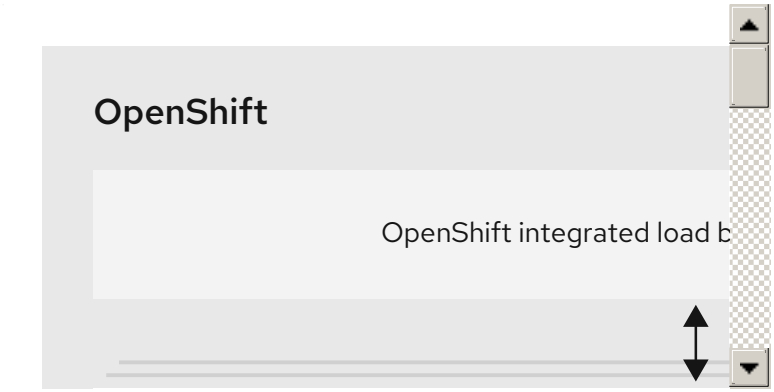


注意

OpenShift Container Platform 支持将集群部署到单个 VMware vCenter 中。不支持在多个 vCenter 上使用机器/机器集部署集群。

12.7.1. 为 vSphere 设置 VMC

您可以在 AWS 托管的 vSphere 集群上安装 OpenShift Container Platform（VMC），以便启用在混合云内部和内部管理应用程序。



在 VMware vSphere 上安装 OpenShift Container Platform 之前，您必须在 VMC 环境中配置多个选项。确定您的 VMC 环境有以下先决条件：

- 创建非专用的、启用了 DHCP 的、NSX-T 网络片段和子网。其他虚拟机（VM）可以托管在子网上，但至少有 8 个 IP 地址可用于 OpenShift Container Platform 部署。
- 配置以下防火墙规则：
  - 在安装主机和软件定义的数据中心（SDDC）管理网络之间的端口 443 的 ANY:ANY 防火墙规则。这可让您在部署过程中上传 Red Hat Enterprise Linux CoreOS（RHCOS）OVA。
  - OpenShift Container Platform 计算网络和 vCenter 间的 HTTPS 防火墙规则。此连接允许 OpenShift Container Platform 与 vCenter 通信，以置备和管理节点、持久性卷声明（PVC）和其他资源。
- 您必须具有以下信息才能部署 OpenShift Container Platform:
  - OpenShift Container Platform 集群名称，如 **vmc-prod-1**。
  - 基本 DNS 名称，如 **companyname.com**。
  - 如果没有使用默认设置，则必须识别 Pod 网络 CIDR 和服务网络 CIDR，默认设置为 **10.128.0.0/14** 和 **172.30.0.0/16**。这些 CIDR 用于 pod 到 pod 和 pod 到服务的通信，且无法在外部访问，但不得与机构中的现有子网重叠。
  - 以下 vCenter 信息：
    - vCenter 主机名、用户名和密码

- 数据中心名称，如 **SDDC-Datacenter**
- 集群名称，如 **Cluster-1**
- 网络名称
- 数据存储名称，如 **WorkloadDatastore**

注意

建议在集群安装完成后将 vSphere 集群移到 VMC **Compute-ResourcePool** 资源池。

- 基于 Linux 的主机作为堡垒部署到 VMC。
  - 堡垒主机可以是 Red Hat Enterprise Linux（RHEL）或其他基于 Linux 的主机，它必须具有互联网连接，并可以将 OVA 上传到 ESXi 主机。
  - 将 OpenShift CLI 工具下载并安装到堡垒主机。
    - **openshift-install** 安装程序
    - OpenShift CLI（**oc**）工具

注意

您不能将 VMware NSX Container Plugin 用于 Kubernetes（NCP），NCP 不使用 NSX 作为 OpenShift SDN。目前 VMC 提供的 NSX 版本与 OpenShift Container Platform 认证的 NCP 版本不兼容。

但是，NSX DHCP 服务会通过全堆栈自动 OpenShift Container Platform 部署来管理虚拟机 IP 管理，节点可以手动或自动置备，可通过 Machine API 与 vSphere 集成。另外，还创建 NSX 防火墙规则，以便启用 OpenShift Container Platform 集群以及堡垒主机和 VMC vSphere 主机间的访问。

12.7.1.1. VMC Sizer 工具

AWS 上的 VMware Cloud 基于 AWS 裸机基础架构构建，这与运行 AWS 原生服务的裸机基础架构相同。当在 AWS 软件定义的数据中心（SDDC）上部署 VMware 云时，您要将这些物理服务器节点以单一租户方式运行 VMware ESXi hypervisor。这意味着其他使用 VMC 的用户无法访问物理基础结构。务必要考虑托管您的虚拟基础架构所需的物理主机数量。

要确定这一点，VMware 在 AWS Sizer 上提供 VMC。使用这个工具，您可以定义要在 VMC 上托管的资源：

- 工作负载类型
- 虚拟机总数
- 规格信息，例如：
  - 存储要求
  - vCPUs
  - vRAM

- 过量使用比例

通过这些信息，sizer 工具可以根据 VMware 最佳实践生成报告，并推荐集群配置和您需要的主机数量。

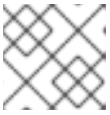
12.7.2. vSphere 先决条件

- 在镜像主机上创建镜像 registry，并获取您的 OpenShift Container Platform 版本的 **imageContentSources** 数据。

重要

由于安装介质位于堡垒主机上，因此请使用该计算机完成所有安装步骤。

- 置备块 registry 存储。如需有关持久性存储的更多信息，请参阅了解持久性存储。
- 查看有关 OpenShift Container Platform 安装和更新流程的详细信息。
- 如果使用防火墙并计划使用遥测（telemetry），您必须将防火墙配置为允许集群需要访问的站点。



注意

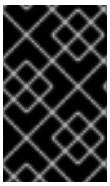
如果您要配置代理，请务必也要查看此站点列表。

12.7.3. 关于在受限网络中安装

在 OpenShift Container Platform 4.6 中，可以执行不需要有效的互联网连接来获取软件组件的安装。受限网络安装可使用安装程序置备的基础架构或用户置备的基础架构完成，具体取决于您要安装集群的云平台。

如果选择在云平台中执行受限网络安装，仍然需要访问其云 API。有些云功能，比如 Amazon Web Service 的 Route 53 DNS 和 IAM 服务，需要访问互联网。根据您的网络，在裸机硬件或 VMware vSphere 上安装时可能需要较少的互联网访问。

要完成受限网络安装，您必须创建一个 registry，镜像 OpenShift Container Platform registry 的内容并包含其安装介质。您可以在堡垒主机上创建此镜像，该主机可同时访问互联网和您的封闭网络，也可以使用满足您的限制条件的其他方法。



重要

由于用户置备安装配置的复杂性，在尝试使用用户置备的基础架构受限网络安装前，请考虑完成标准用户置备的基础架构安装。通过完成此测试安装，您可以更轻松地隔离和排查您在受限网络中安装时可能出现的问题。

12.7.3.1. 其他限制

受限网络中的集群还有以下额外限制：

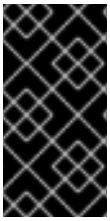
- **ClusterVersion** 状态包含一个 **Unable to retrieve available updates** 错误。
- 默认情况下，您无法使用 Developer Catalog 的内容，因为您无法访问所需的镜像流标签。

12.7.4. OpenShift Container Platform 的互联网访问

在 OpenShift Container Platform 4.6 中，您需要访问互联网来获得用来安装集群的镜像。

您必须具有以下互联网访问权限：

- 访问 OpenShift Cluster Manager 以下载安装程序并执行订阅管理。如果集群可以访问互联网，并且没有禁用 Telemetry，该服务会自动授权您的集群。
- 访问 Quay.io，以获取安装集群所需的软件包。
- 获取执行集群更新所需的软件包。



重要

如果您的集群无法直接访问互联网，则可以在置备的某些类基础架构上执行受限网络安装。在此过程中，您要下载所需的内容，并使用它在镜像 registry（mirror registry） 中填充安装集群并生成安装程序所需的软件包。对于某些安装类型，集群要安装到的环境不需要访问互联网。在更新集群之前，要更新 registry 镜像系统中的内容。

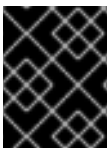
12.7.5. VMware vSphere 基础架构要求

您必须在满足您使用的组件要求的 VMware vSphere 版本 6 或 7 实例上安装 OpenShift Container Platform 集群。

表 12.62. VMware 组件支持的最低 vSphere 版本

组件	最低支持版本	描述
虚拟机监控程序	vSphere 6.5 及之后的版本 13	此版本是 Red Hat Enterprise Linux CoreOS(RHCOS)支持的最低版本。请查看 Red Hat Enterprise Linux 8 支持的管理程序列表。
使用 in-tree 驱动程序存储	vSphere 6.5 及之后的版本	此插件使用 OpenShift Container Platform 中包含的 vSphere 的树内存储驱动程序创建 vSphere 存储。

如果您使用 vSphere 版本 6.5 实例，请在安装 OpenShift Container Platform 前考虑升级到 6.7U3 或 7.0。



重要

您必须确保在安装 OpenShift Container Platform 前同步 ESXi 主机上的时间。请参阅 VMware 文档中的编辑主机时间配置。

12.7.6. 具有用户置备基础架构的集群的机器要求

对于含有用户置备的基础架构的集群，您必须部署所有所需的机器。

12.7.6.1. 所需的机器

最小的 OpenShift Container Platform 集群需要下列主机：

- 一个临时 bootstrap 机器
- 三台 control plane 或 master 机器
- 至少两台计算机器，也称为 worker 机器。

注意

集群要求 bootstrap 机器在三台 control plane 机器上部署 OpenShift Container Platform 集群。您可在安装集群后删除 bootstrap 机器。

重要

要保持集群的高可用性，请将独立的物理主机用于这些集群机器。

bootstrap 和 control plane 机器必须使用 Red Hat Enterprise Linux CoreOS (RHCOS) 作为操作系统。但是，计算机器可以在 Red Hat Enterprise Linux CoreOS(RHCOS)或 Red Hat Enterprise Linux(RHEL)7.9 之间进行选择。

请注意，RHCOS 基于 Red Hat Enterprise Linux（RHEL） 8，并继承其所有硬件认证和要求。请查看Red Hat Enterprise Linux 技术功能及限制 。

12.7.6.2. 网络连接要求

所有 Red Hat Enterprise Linux CoreOS (RHCOS) 机器在启动过程中需要 **initramfs** 中的网络从 Machine Config Server 获取 Ignition 配置文件。在初次启动过程中，需要一个 DHCP 服务器或设置了静态 IP 地址来建立网络连接，以下载它们的 Ignition 配置文件。另外，集群中的每个 OpenShift Container Platform 节点都必须有权访问网络时间协议（NTP）服务器。如果 DHCP 服务器提供 NTP 服务器信息，Red Hat Enterprise Linux CoreOS（RHCOS）机器上的 chrony 时间服务会读取信息，并可与 NTP 服务器同步时钟。

12.7.6.3. 最低资源要求

每台集群机器都必须满足以下最低要求：

表 12.63. 最低资源要求

机器	操作系统	vCPU [1]	虚拟内存	存储	IOPS [2]
bootstrap	RHCOS	4	16 GB	100 GB	300
Control plane	RHCOS	4	16 GB	100 GB	300
Compute	RHCOS 或 RHEL 7.9	2	8 GB	100 GB	300

1. 当未启用并发多线程 (SMT) 或超线程时，一个 vCPU 相当于一个物理内核。启用后，使用以下公式来计算对应的比例：（每个内核数的线程）× sockets = vCPU。
2. OpenShift Container Platform 和 Kubernetes 对磁盘性能非常敏感，建议使用更快的存储速度，特别是 control plane 节点上需要 10 ms p99 fsync 持续时间的 etcd。请注意，在许多云平台上，存储大小和 IOPS 可一起扩展，因此您可能需要过度分配存储卷来获取足够的性能。

12.7.6.4. 证书签名请求管理

在使用您置备的基础架构时，集群只能有限地访问自动机器管理，因此您必须提供一种在安装后批准集群证书签名请求 (CSR) 的机制。**kube-controller-manager** 只能批准 kubelet 客户端 CSR。 **machine-approver** 无法保证使用 kubelet 凭证请求的提供证书的有效性，因为它不能确认是正确的机器发出了该请求。您必须决定并实施一种方法，以验证 kubelet 提供证书请求的有效性并进行批准。

12.7.7. 创建用户置备的基础架构

在部署采用用户置备的基础架构的 OpenShift Container Platform 集群前，您必须创建底层基础架构。

先决条件

- 在为集群创建支持基础架构之前，请参阅OpenShift Container Platform 4.x Tested Integrations页。

流程

1. 在每个节点上配置 DHCP 或设置静态 IP 地址。
2. 提供所需的负载均衡器。
3. 配置机器的端口。
4. 配置 DNS。
5. 确保网络可以正常工作。

12.7.7.1. 用户置备的基础架构对网络的要求

所有 Red Hat Enterprise Linux CoreOS（RHCOS）机器在启动过程中需要 **initramfs** 中的网络从机器配置服务器获取 Ignition 配置。

在初次启动过程中，需要一个 DHCP 服务器或集群中的每个机器都设置了静态 IP 地址来建立网络连接，以下载它们的 Ignition 配置文件。

建议您使用 DHCP 服务器为集群进行长期机器管理。确保 DHCP 服务器已配置为向集群机器提供持久 IP 地址和主机名。

Kubernetes API 服务器必须能够解析集群机器的节点名称。如果 API 服务器和 worker 节点位于不同的区域中，您可以配置默认 DNS 搜索区域，以便 API 服务器能够解析节点名称。另一种支持的方法是始终在节点对象和所有 DNS 请求中使用完全限定域名来指代主机。

您必须配置机器间的网络连接，以便集群组件进行通信。每台机器都必须能够解析集群中所有其他机器的主机名。

表 12.64. 所有机器到所有机器

协议	端口	描述
ICMP	N/A	网络可访问性测试
TCP	**1936**	指标

协议	端口	描述
	9000-9999	主机级别的服务，包括端口 **9100-9101** 上的节点导出器和端口 **9099** 上的 Cluster Version Operator。
	10250-10259	Kubernetes 保留的默认端口
	10256	openshift-sdn
UDP	**4789**	VXLAN 和 Geneve
	6081	VXLAN 和 Geneve
	9000-9999	主机级别的服务，包括端口 **9100-9101** 上的节点导出器。
TCP/UDP	**30000-32767**	Kubernetes 节点端口

表 12.65. 要通过控制平面的所有机器

协议	端口	描述
TCP	**6443**	Kubernetes API

表 12.66. control plane 机器到 control plane 机器

协议	端口	描述
TCP	**2379-2380**	etcd 服务器和对等端口

网络拓扑要求

您为集群置备的基础架构必须满足下列网络拓扑要求。

重要

OpenShift Container Platform 要求所有节点都能访问互联网，以便为平台容器提取镜像并向红帽提供遥测数据。

负载均衡器

在安装 OpenShift Container Platform 前，您必须置备两个满足以下要求的负载均衡器：

1. **API 负载均衡器**：提供一个通用端点，供用户（包括人和机器）与平台交互和配置。配置以下条件：
   - 只适用于第 4 层负载均衡。这可被称为 Raw TCP、SSL Passthrough 或者 SSL 桥接模式。如果使用 SSL Bridge 模式，必须为 API 路由启用 Server Name Indication（SNI）。
   - 无状态负载平衡算法。这些选项根据负载均衡器的实现而有所不同。

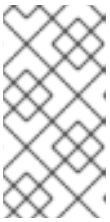
重要

不要为 API 负载均衡器配置会话持久性。

在负载均衡器的前端和后台配置以下端口：

表 12.67. API 负载均衡器

端口	后端机器（池成员）	内部	外部	描述
6443	Bootstrap 和 control plane.bootstrap 机器初始化集群 control plane 后，您要从负载均衡器中删除 bootstrap 机器。您必须为 API 服务器健康检查探测配置 **/readyz** 端点。	X	X	Kubernetes API 服务器
22623	Bootstrap 和 control plane.bootstrap 机器初始化集群 control plane 后，您要从负载均衡器中删除 bootstrap 机器。	X		机器配置服务器



注意

负载均衡器必须配置为，从 API 服务器关闭 **/readyz** 端点到从池中删除 API 服务器实例时最多需要 30 秒。在 **/readyz** 返回错误或处于健康状态后的时间范围内，端点必须被删除或添加。每 5 秒或 10 秒探测一次，有两个成功请求处于健康状态，三个成为不健康的请求经过测试。

2. **应用程序入口负载均衡器**:提供来自集群外部的应用程序流量流量的 Ingress 点。配置以下条件：

   - 只适用于第 4 层负载均衡。这可被称为 Raw TCP、SSL Passthrough 或者 SSL 桥接模式。如果使用 SSL Bridge 模式，您必须为 Ingress 路由启用Server Name Indication（SNI）。
   - 建议根据可用选项以及平台上托管的应用程序类型，使用基于连接的或者基于会话的持久性。

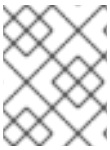
   在负载均衡器的前端和后台配置以下端口：

   表 12.68. 应用程序入口负载均衡器

端口	后端机器（池成员）	内部	外部	描述
443	默认运行入口路由器 Pod、计算或 worker 的机器。	X	X	HTTPS 流量
80	默认运行入口路由器 Pod、计算或 worker 的机器。	X	X	HTTP 流量

提示

如果负载均衡器可以看到客户端的真实 IP 地址，启用基于 IP 的会话持久性可提高使用端到端 TLS 加密的应用程序的性能。



注意

OpenShift Container Platform 集群需要正确配置入口路由器。control plane 初始化后，您必须配置入口路由器。

NTP 配置

OpenShift Container Platform 集群默认配置为使用公共网络时间协议（NTP）服务器。如果要使用本地企业 NTP 服务器，或者集群部署在断开连接的网络中，您可以将集群配置为使用特定的时间服务器。如需更多信息，请参阅*配置 chrony 时间服务*的文档。

如果 DHCP 服务器提供 NTP 服务器信息，Red Hat Enterprise Linux CoreOS（RHCOS）机器上的 chrony 时间服务会读取信息，并可与 NTP 服务器同步时钟。

12.7.7.2. 用户置备 DNS 要求

DNS 用于名称解析和反向名称解析。DNS A/AAAA 或 CNAME 记录用于名称解析，PTR 记录用于反向解析名称。反向记录很重要，因为 Red Hat Enterprise Linux CoreOS（RHCOS）使用反向记录为所有节点设置主机名。另外，反向记录用于生成 OpenShift Container Platform 需要操作的证书签名请求（CSR）。

采用用户置备的基础架构的 OpenShift Container Platform 集群需要以下 DNS 记录。在每一记录中，**<cluster_name>** 是集群名称，**<base_domain>** 则是您在 **install-config.yaml** 文件中指定的集群基域。完整的 DNS 记录采用如下格式: **<component>.<cluster_name>.<base_domain>.**。

表 12.69. 所需的 DNS 记录

组件	记录	描述
Kubernetes API	**api.<cluster_name>. <base_domain>.**	添加 DNS A/AAAA 或 CNAME 记录，以及 DNS PTR 记录，以识别 control plane 机器的负载均衡器。这些记录必须由集群外的客户端以及集群中的所有节点解析。
	api-int.<cluster_name>. <base_domain>.	添加 DNS A/AAAA 或 CNAME 记录，以及 DNS PTR 记录，以识别 control plane 机器的负载均衡器。这些记录必须可以从集群中的所有节点解析。  **重要** API 服务器必须能够根据在 Kubernetes 中记录的主机名解析 worker 节点。如果 API 服务器无法解析节点名称，则代理的 API 调用会失败，且您无法从 pod 检索日志。
Routes	***.apps.<cluster_name>. <base_domain>.**	添加通配符 DNS A/AAAA 或 CNAME 记录，指向以运行入口路由器 Pod 的机器（默认为 worker 节点）为目标的负载均衡器。这些记录必须由集群外的客户端以及集群中的所有节点解析。
bootstrap	**bootstrap.<cluster_name>. <base_domain>.**	添加 DNS A/AAAA 或 CNAME 记录，以及 DNS PTR 记录来识别 bootstrap 机器。这些记录必须由集群中的节点解析。

组件	记录	描述
Master 主机	**<master><n>. <cluster_name>. <base_domain>.**	DNS A/AAAA 或 CNAME 记录，以识别 control plane 节点（也称为 master 节点）的每台机器。这些记录必须由集群中的节点解析。
Worker 主机	**<worker><n>. <cluster_name>. <base_domain>.**	添加 DNS A/AAAA 或 CNAME 记录，以识别 worker 节点的每台机器。这些记录必须由集群中的节点解析。

提示

您可以使用 **nslookup <hostname>** 命令来验证名称解析。您可以使用 **dig -x <ip_address>** 命令来验证 PTR 记录的反向名称解析。

下面的 BIND 区文件的例子展示了关于名字解析的 A 记录的例子。这个示例的目的是显示所需的记录。这个示例不是为选择一个名称解析服务提供建议。

例 12.13. DNS 区数据库示例

```
$TTL 1W
@ IN SOA ns1.example.com. root (
    2019070700 ; serial
    3H  ; refresh (3 hours)
    30M  ; retry (30 minutes)
    2W  ; expiry (2 weeks)
    1W )  ; minimum (1 week)
 IN NS ns1.example.com.
 IN MX 10 smtp.example.com.
;
;
ns1 IN A 192.168.1.5
smtp IN A 192.168.1.5
;
helper IN A 192.168.1.5
helper.ocp4 IN A 192.168.1.5
;
; The api identifies the IP of your load balancer.
api.ocp4  IN A 192.168.1.5
api-int.ocp4  IN A 192.168.1.5
;
; The wildcard also identifies the load balancer.
*.apps.ocp4  IN A 192.168.1.5
;
; Create an entry for the bootstrap host.
bootstrap.ocp4 IN A 192.168.1.96
;
; Create entries for the master hosts.
master0.ocp4  IN A 192.168.1.97
master1.ocp4  IN A 192.168.1.98
master2.ocp4  IN A 192.168.1.99
;
; Create entries for the worker hosts.
```

```
worker0.ocp4  IN A 192.168.1.11
worker1.ocp4  IN A 192.168.1.7
;
;EOF
```

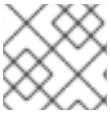
下面的 BIND 区文件示例显示了反向名字解析的 PTR 记录示例。

例 12.14. 反向记录的 DNS 区数据库示例

```
$TTL 1W
@ IN SOA ns1.example.com. root (
   2019070700 ; serial
   3H  ; refresh (3 hours)
   30M  ; retry (30 minutes)
   2W  ; expiry (2 weeks)
   1W )  ; minimum (1 week)
 IN NS ns1.example.com.
;
; The syntax is "last octet" and the host must have an FQDN
; with a trailing dot.
97 IN PTR master0.ocp4.example.com.
98 IN PTR master1.ocp4.example.com.
99 IN PTR master2.ocp4.example.com.
;
96 IN PTR bootstrap.ocp4.example.com.
;
5 IN PTR api.ocp4.example.com.
5 IN PTR api-int.ocp4.example.com.
;
11 IN PTR worker0.ocp4.example.com.
7 IN PTR worker1.ocp4.example.com.
;
;EOF
```

12.7.8. 生成 SSH 私钥并将其添加到代理中

如果要在集群上执行安装调试或灾难恢复，则必须为 **ssh-agent** 和安装程序提供 SSH 密钥。您可以使用此密钥访问公共集群中的 bootstrap 机器来排除安装问题。



注意

在生产环境中，您需要进行灾难恢复和调试。

您可以使用此密钥以 **core** 用户身份通过 SSH 连接到 master 节点。在部署集群时，此密钥会添加到 **core** 用户的 **~/.ssh/authorized_keys** 列表中。

注意

您必须使用一个本地密钥，而不要使用在特定平台上配置的密钥，如 AWS 密钥对。

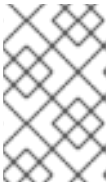
流程

1. 如果还没有为计算机上免密码身份验证而配置的 SSH 密钥，请创建一个。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ ssh-keygen -t ed25519 -N '' \
  -f <path>/<file_name> 1
```

1 指定新 SSH 密钥的路径和文件名，如 **~/.ssh/id_rsa**。如果您已有密钥对，请确保您的公钥位于 **~/.ssh** 目录中。

运行此命令会在指定的位置生成不需要密码的 SSH 密钥。



注意

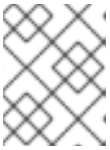
如果您计划在 **x86_64** 架构中安装使用 FIPS 验证的/Modules in Process 加密库的 OpenShift Container Platform 集群，不要创建使用 **ed25519** 算法的密钥。反之，创建一个使用 **rsa** 或 **ecdsa** 算法的密钥。

2. 作为后台任务启动 **ssh-agent** 进程：

```
$ eval "$(ssh-agent -s)"
```

输出示例

```
Agent pid 31874
```



注意

如果您的集群采用 FIPS 模式，则只使用 FIPS 兼容算法来生成 SSH 密钥。密钥必须是 RSA 或 ECDSA。

3. 将 SSH 私钥添加到 **ssh-agent**：

```
$ ssh-add <path>/<file_name> 1
```

输出示例

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

1 指定 SSH 私钥的路径和文件名，如 **~/.ssh/id_rsa**

后续步骤

- 在安装 OpenShift Container Platform 时，为安装程序提供 SSH 公钥。如果在您置备的基础架构上安装集群，您必须将此密钥提供给集群的机器。

12.7.9. 手动创建安装配置文件

对于使用用户置备的基础架构的 OpenShift Container Platform 安装，您必须手动生成安装配置文件。

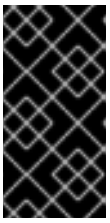
先决条件

- 获取 OpenShift Container Platform 安装程序和集群的访问令牌。
- 获取命令输出中的 **imageContentSources** 部分来镜像存储库。
- 获取您的镜像 registry 的证书内容。

流程

1. 创建用来存储您所需的安装资产的安装目录：

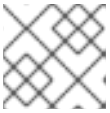
```
$ mkdir <installation_directory>
```



重要

您必须创建目录。一些安装信息，如 bootstrap X.509 证书，有较短的过期间隔，因此不要重复使用安装目录。如果要重复使用另一个集群安装中的个别文件，可以将其复制到您的目录中。但是，一些安装数据的文件名可能会在发行版本之间有所改变。从 OpenShift Container Platform 老版本中复制安装文件时要格外小心。

2. 自定义以下 **install-config.yaml** 文件模板，并将它保存到 **<installation_directory>** 中。



注意

此配置文件必须命名为 **install-config.yaml**。

- 除非使用 RHCOS 默认信任的 registry，如 **docker.io**，否则必须在 **additionalTrustBundle** 部分中提供镜像存储库的证书内容。在大多数情况下，必须为您的镜像提供证书。
- 您必须包含命令输出中的 **imageContentSources** 部分，才能镜像存储库。

3. 备份 **install-config.yaml** 文件，以便用于安装多个集群。

重要

install-config.yaml 文件会在安装过程的下一步骤中消耗掉。现在必须备份它。

12.7.9.1. VMware vSphere install-config.yaml 文件示例

您可以自定义 **install-config.yaml** 文件，以指定有关 OpenShift Container Platform 集群平台的更多信息，或修改所需参数的值。

```
apiVersion: v1
baseDomain: example.com (1)
compute:
- hyperthreading: Enabled (2) (3)
  name: worker
  replicas: 0 (4)
controlPlane:
  hyperthreading: Enabled (5) (6)
  name: master
  replicas: 3 (7)
```

```
metadata:
  name: test 8
platform:
  vsphere:
    vcenter: your.vcenter.server 9
    username: username 10
    password: password 11
    datacenter: datacenter 12
    defaultDatastore: datastore 13
    folder: "/<datacenter_name>/vm/<folder_name>/<subfolder_name>" 14
fips: false 15
pullSecret: '{"auths":{"<local_registry>": {"auth": "<credentials>","email": "you@example.com"}}}' 16
sshKey: 'ssh-ed25519 AAAA...' 17
additionalTrustBundle: | 18
  -----BEGIN CERTIFICATE-----
  ZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZ
  -----END CERTIFICATE-----
imageContentSources: 19
- mirrors:
  - <local_registry>/<local_repository_name>/release
  source: quay.io/openshift-release-dev/ocp-release
- mirrors:
  - <local_registry>/<local_repository_name>/release
  source: quay.io/openshift-release-dev/ocp-v4.0-art-dev
```

1 集群的基域。所有 DNS 记录都必须是这个基域的子域，并包含集群名称。

2 5 **controlPlane** 部分是一个单映射，但 compute 部分是一系列映射。为满足不同数据结构的要求，**compute** 部分的第一行必须以连字符 **-** 开头，**controlPlane** 部分的第一行则不可以连字符开头。虽然这两个部分目前都定义单个机器池，但未来的 OpenShift Container Platform 版本可能会支持在安装过程中定义多个计算池。只使用一个 control plane 池。

3 6 是否要启用或禁用并发多线程或**超线程**。默认情况下，启用并发多线程以提高机器内核的性能。您可以通过将参数值设为 **Disabled** 来禁用。如果您在某些集群机器上禁用并发多线程，则必须在所有集群机器上禁用。

重要

如果禁用并发多线程，请确保在容量规划时考虑到机器性能可能会显著降低的问题。如果您禁用并发多线程，则计算机必须至少使用 8 个 CPU 和 32GB RAM。

4 **replicas** 参数的值必须设置为 **0**。此参数控制集群为您创建和管理的 worker 数量，使用用户置备的基础架构时集群不会执行这些功能。在完成 OpenShift Container Platform 安装前，您必须手动为集群部署 worker 机器。

7 您添加到集群的 control plane 机器数量。由于集群将这个值用作集群中 etcd 端点的数量，因此该值必须与您部署的 control plane 机器数量匹配。

8 您在 DNS 记录中指定的集群名称。

9 vCenter 服务器的完全限定主机名或 IP 地址。

10 用于访问服务器的用户名。此用户必须至少具有 vSphere 中 静态或动态持久性卷置备 所需的角色和权限。

11 与 vSphere 用户关联的密码。

12 vSphere 数据中心。

13 要使用的默认 vSphere 数据存储。

14 可选：对于安装程序置备的基础架构，安装程序创建虚拟机的现有文件夹的绝对路径，如 **/<datacenter_name>/vm/<folder_name>/<subfolder_name>**。如果没有提供这个值，安装程序会在数据中心虚拟机文件夹中创建一个顶层文件夹，其名称为基础架构 ID。如果您为集群提供基础架构,请省略此参数。

15 是否启用或禁用 FIPS 模式。默认情况下不启用 FIPS 模式。如果启用了 FIPS 模式，运行 OpenShift Container Platform 的 Red Hat Enterprise Linux CoreOS (RHCOS) 机器会绕过默认的 Kubernetes 加密套件，并使用由 RHCOS 提供的加密模块。

重要

只有在 **x86_64** 架构中的 OpenShift Container Platform 部署支持 FIPS 验证的/Modules in Process 加密库。

16 对于 **<local_registry>**，请指定 registry 域名，以及您的镜像 registry 用来提供内容的可选端口。例如： **registry.example.com** 或者 **registry.example.com:5000**。使用 **<credentials>** 为您生成的镜像 registry 指定 base64 编码的用户名和密码。

17 Red Hat Enterprise Linux CoreOS (RHCOS) 中 **core** 用户的默认 SSH 密钥的公钥部分。

注意

对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 **ssh-agent** 进程使用的 SSH 密钥。

18 提供用于镜像 registry 的证书文件内容。

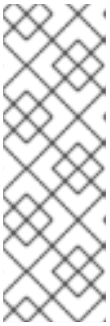
19 提供命令输出中的 **imageContentSources** 部分来镜像存储库。

12.7.9.2. 在安装过程中配置集群范围代理

生产环境可能会拒绝直接访问互联网，而是提供 HTTP 或 HTTPS 代理。您可以通过在 **install-config.yaml** 文件中配置代理设置，将新的 OpenShift Container Platform 集群配置为使用代理。

先决条件

- 您有一个现有的 **install-config.yaml** 文件。
- 您检查了集群需要访问的站点，并决定是否需要绕过代理。默认情况下代理所有集群出口流量，包括对托管云供应商 API 的调用。您需要将站点添加到 **Proxy** 对象的 **spec.noProxy** 字段来绕过代理。



注意

Proxy 对象 **status.noProxy** 字段使用安装配置中的 **networking.machineNetwork[].cidr**、**networking.clusterNetwork[].cidr** 和 **networking.serviceNetwork[]** 字段的值填充。

对于在 Amazon Web Services(AWS)、Google Cloud Platform(GCP)、Microsoft Azure 和 Red Hat OpenStack Platform(RHOSP)上安装, **Proxy** 对象 **status.noProxy** 字段也会使用实例元数据端点填充(**169.254.169.254**)。

流程

1. 编辑 **install-config.yaml** 文件并添加代理设置。例如：

```
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: example.com 3
additionalTrustBundle: | 4
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
...
```

1. 用于创建集群外 HTTP 连接的代理 URL。URL 必须是 **http**。
2. 用于创建集群外 HTTPS 连接的代理 URL。
3. 要排除在代理中的目标域名、IP 地址或其他网络 CIDR 的逗号分隔列表。在域前面加 **.** 来仅匹配子域。例如： **.y.com** 匹配 **x.y.com**，但不匹配 **y.com**。使用 * 绕过所有目的地的代理。您必须包含 vCenter 的 IP 地址以及用于其机器的 IP 范围。
4. 如果提供，安装程序会在 **openshift-config 命名空间中生成名为 user-ca- bundle** 的配置映射来保存额外的 CA 证书。如果您提供 **additionalTrustBundle** 和至少一个代理设置，则 **Proxy** 对象会被配置为引用 **trustedCA** 字段中的 **user-ca-bundle** 配置映射。然后，Cluster Network Operator 会创建一个 **trusted-ca-bundle** 配置映射，该配置映射将为 **trustedCA** 参数指定的内容与 RHCOS 信任捆绑包合并。 **additionalTrustBundle** 字段是必需的，除非代理的身份证书由来自 RHCOS 信任捆绑包的颁发机构签名。

注意

安装程序不支持代理的 **readinessEndpoints** 字段。

2. 保存该文件，并在安装 OpenShift Container Platform 时引用。

安装程序会创建一个名为 **cluster** 的集群范围代理，该代理使用提供的 **install-config.yaml** 文件中的代理设置。如果没有提供代理设置，仍然会创建一个 **cluster Proxy** 对象，但它会有一个空 **spec**。

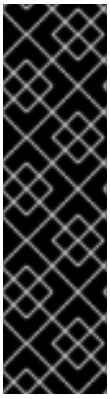
注意

只支持名为 **cluster** 的 **Proxy** 对象，且无法创建额外的代理。

12.7.10. 创建 Kubernetes 清单和 Ignition 配置文件

由于您必须修改一些集群定义文件并要手动启动集群机器，因此您必须生成 Kubernetes 清单和 Ignition 配置文件，集群需要这两项来创建其机器。

安装配置文件转换为 Kubernetes 清单。清单嵌套到 Ignition 配置文件中，稍后用于创建集群。



重要

- 安装程序生成的 Ignition 配置文件包含在 24 小时后过期的证书，然后在过期时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外情况是，您需要手动批准待处理的 **node-bootstrapper** 证书签名请求（CSR）来恢复 kubelet 证书。如需更多信息，请参阅*从过期的 control plane 证书中恢复*的文档。
- 建议您在生成 12 小时后使用 Ignition 配置文件，因为集群安装后 24 小时证书从 16 小时轮转至 22 小时。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中运行证书更新时避免安装失败。

先决条件

- 已获得 OpenShift Container Platform 安装程序。对于受限网络安装，这些文件位于您的堡垒主机上。
- 已创建 **install-config.yaml** 安装配置文件。

流程

1. 切换到包含安装程序的目录，并为集群生成 Kubernetes 清单：

```
$ ./openshift-install create manifests --dir <installation_directory> 1
```

1 对于 **<installation_directory>**，请指定含有您创建的 **install-config.yaml** 文件的安装目录。

2. 删除定义 control plane 机器的 Kubernetes 清单文件以及计算机器集：

```
$ rm -f openshift/99_openshift-cluster-api_master-machines-*.yaml openshift/99_openshift-cluster-api_worker-machineset-*.yaml
```

由于您要自行创建和管理这些资源，因此不必初始化这些资源。

- 您可以使用机器 API 来保留机器集文件来创建计算机器，但您必须更新对其的引用，以匹配您的环境。

3. 检查 **<installation_directory>/manifests/cluster-scheduler-02-config.yml** Kubernetes 清单文件中的 **mastersSchedulable** 参数是否已设置为 **false**。此设置可防止在 control plane 机器上调度 pod:

   a. 打开 **<installation_directory>/manifests/cluster-scheduler-02-config.yml** 文件。

   b. 找到 **mastersSchedulable** 参数并确保它被设置为 **false**。

   c. 保存并退出文件。

4. 要创建 Ignition 配置文件，从包含安装程序的目录运行以下命令：

```
$ ./openshift-install create ignition-configs --dir <installation_directory> 1
```

1 对于 **<installation_directory>**，请指定相同的安装目录。

该目录中将生成以下文件：

```
.
├── auth
│   ├── kubeadmin-password
│   └── kubeconfig
├── bootstrap.ign
├── master.ign
├── metadata.json
└── worker.ign
```

12.7.11. 提取基础架构名称

Ignition 配置文件包含一个唯一的集群标识符,您可以使用它在 VMware Cloud 中唯一地标识您的集群。如果计划使用集群标识符作为虚拟机文件夹的名称,您必须提取它。

先决条件

- 获取 OpenShift Container Platform 安装程序以及集群的 pull secret。
- 已为集群生成 Ignition 配置文件。
- 安装了 **jq** 软件包。

流程

- 要从 Ignition 配置文件元数据中提取和查看基础架构名称，请运行以下命令：

```
$ jq -r .infraID <installation_directory>/metadata.json 1
```

1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

输出示例

```
openshift-vw9j6 1
```

1 此命令的输出是您的集群名称和随机字符串。

12.7.12. 在 vSphere 中创建 Red Hat Enterprise Linux CoreOS (RHCOS) 机器

在 VMware vSphere 上安装包含用户置备基础架构的集群前，您必须在 vSphere 主机上创建 RHCOS 机器供其使用。

先决条件

- 已获取集群的 Ignition 配置文件。
- 您可以从计算机访问 HTTP 服务器，并且您创建的机器也可以访问该服务器。
- 您已创建了 vSphere 集群。

流程

1. 将名为 **<installation_directory>/bootstrap.ign** 的 bootstrap Ignition 配置文件上传到 HTTP 服务器，该配置文件是由安装程序创建的。记下此文件的 URL。
2. 将 bootstrap 节点的以下辅助 Ignition 配置文件保存到计算机中，存为 **<installation_directory>/merge-bootstrap.ign** ：

```
{
  "ignition": {
    "config": {
      "merge": [
        {
          "source": "<bootstrap_ignition_config_url>", 1
          "verification": {}
        }
      ]
    },
    "timeouts": {},
    "version": "3.1.0"
  },
  "networkd": {},
  "passwd": {},
  "storage": {},
  "systemd": {}
}
```

1 指定您托管的 bootstrap Ignition 配置文件的 URL。

为 bootstrap 机器创建虚拟机 (VM) 时，您要使用此 Ignition 配置文件。

3. 找到安装程序创建的以下 Ignition 配置文件：
   - **<installation_directory>/master.ign**
   - **<installation_directory>/worker.ign**
   - **<installation_directory>/merge-bootstrap.ign**
4. 将 Ignition 配置文件转换为 Base64 编码。在此流程中，您必须将这些文件添加到虚拟机中的额外配置参数 **guestinfo.ignition.config.data** 中。
   例如，如果您使用 Linux 操作系统，可以使用 **base64** 命令来编码这些文件。

```
$ base64 -w0 <installation_directory>/master.ign > <installation_directory>/master.64
```

```
$ base64 -w0 <installation_directory>/worker.ign > <installation_directory>/worker.64
```

```
$ base64 -w0 <installation_directory>/merge-bootstrap.ign > <installation_directory>/merge-bootstrap.64
```

重要

如果您计划在安装完成后在集群中添加更多计算机器，请不要删除这些文件。

5. 获取 RHCOS OVA 镜像。镜像位于 RHCOS 镜像镜像页面。

重要

RHCOS 镜像可能不会随着 OpenShift Container Platform 的每一发行版本都有改变。您必须下载一个最高版本的镜像，其版本号应小于或等于您安装的 OpenShift Container Platform 版本。如果可用，请使用与 OpenShift Container Platform 版本匹配的镜像版本。

文件名包含 OpenShift Container Platform 版本号，格式为 **rhcos-vmware.<architecture>.ova**。

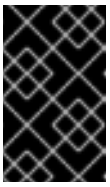
6. 在 vSphere 客户端中，在数据中心中创建一个文件夹来存储您的虚拟机。
   a. 点击 **VMs and Templates** 视图。
   b. 右键点击您的数据中心名称。
   c. 点击 **New Folder → New VM and Template Folder**。
   d. 在显示的窗口中输入文件夹名称。如果您没有在 **install-config.yaml** 文件中指定现有文件夹，请创建一个文件夹，其名称与基础架构 ID 相同。您可以使用这个文件夹名称，因此 vCenter 会在适当的位置为 Workspace 配置动态置备存储。
7. 在 vSphere 客户端中，为 OVA 镜像创建一个模板，然后根据需要克隆模板。

注意

在以下步骤中，您将创建一个模板，然后克隆所有集群机器的模板。然后，在置备虚拟机时，为该克隆的机器类型提供 Ignition 配置文件的位置。

   a. 在 **Hosts and Clusters** 选项卡中，右键点击您的集群名称并选择 **Deploy OVF Template**。
   b. 在 **Select an OVF** 选项卡中，指定您下载的 RHCOS OVA 文件的名称。
   c. 在 **Select a name and folder** 选项卡中，为您的模板设置 **虚拟机名称**，如 **Template-RHCOS**。点击 vSphere 集群的名称并选择您在上一步中创建的文件夹。
   d. 在 **Select a compute resource** 选项卡中，点击您的 vSphere 集群名称。
   e. 在 **Select storage** 选项卡中，配置虚拟机的存储选项。
      - 根据您的存储要求，选择**Thin Provision** 或 **Thick Provision**。
      - 选择您在 **install-config.yaml** 文件中指定的数据存储。
   f. 在 **Select network** 选项卡中，指定您为集群配置的网络（如果可用）。

g. 在创建 OVF 模板时，请不要在 **Customize template** 选项卡上指定值，或者不要再配置模板。



重要

不要启动原始虚拟机模板。VM 模板必须保持关闭状态，必须为新的 RHCOS 机器克隆。启动虚拟机模板会将虚拟机模板配置为平台上的虚拟机，这样可防止它被用作计算机集可以应用配置的模板。

8. 部署模板后，为集群中的机器部署虚拟机。

a. 右键点击模板的名称，再点击 **Clone → Clone to Virtual Machine**。

b. 在 **Select a name and folder**选项卡中，指定虚拟机的名称。名称中可以包括机器类型，如 **control-plane-0** 或 **compute-1**。

c. 在 **Select a name and folder**选项卡中，选择您为集群创建的文件夹名称。

d. 在 **Select a compute resource** 选项卡中，选择数据中心中的主机名称。
对于 bootstrap 机器，指定您托管的 bootstrap Ignition 配置文件的 URL。

e. 可选：在 **Select storage** 选项卡中，自定义存储选项。

f. 在 **Select clone options** 中，选择 **Customize this virtual machine's hardware**。

g. 在 **Customize hardware** 选项卡中，点击 **VM Options → Advanced**。

- 可选：覆盖 vSphere 中的默认 DHCP 网络。启用静态 IP 网络：

i. 设置静态 IP 配置：

```
$ export IPCFG="ip=<ip>::<gateway>:<netmask>:<hostname>:<iface>:none
nameserver=srv1 [nameserver=srv2 [nameserver=srv3 [...]]]"
```

示例命令

```
$ export IPCFG="ip=192.168.100.101::192.168.100.254:255.255.255.0:::none
nameserver=8.8.8.8"
```

ii. 在从 vSphere 中的 OVA 引导虚拟机前，设置 **guestinfo.afterburn.initrd.network-kargs** 属性：

```
$ govc vm.change -vm "<vm_name>" -e "guestinfo.afterburn.initrd.network-
kargs=${IPCFG}"
```

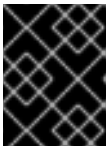
- 可选：在出现集群性能问题时，从 **Latency Sensitivity** 列表中选择 **High**。确定虚拟机的 CPU 和内存保留有以下值：
  - 内存保留值必须等于其配置的内存大小。
  - CPU 保留值必须至少是低延迟虚拟 CPU 的数量，乘以测量的物理 CPU 速度。
- 点击 **Edit Configuration**，然后在 **Configuration Parameters** 窗口中点击 **Add Configuration Params**。定义以下参数名称和值：

- **guestinfo.ignition.config.data** ：找到您在此流程中创建的 base-64 编码文件，并粘贴此机器类型的 base64 编码 Ignition 配置文件的内容。
- **guestinfo.ignition.config.data.encoding**：指定 **base64**。
- **disk.EnableUUID**：指定 **TRUE**。

h. 在 **Customize hardware** 选项卡的 **Virtual Hardware** 面板中，根据需要修改指定的值。确保 RAM、CPU 和磁盘存储的数量满足机器类型的最低要求。

i. 完成配置并打开虚拟机电源。

9. 对于每台机器，按照前面的步骤为集群创建其余的机器。



重要

此刻您必须创建 bootstrap 和 control plane 机器。由于计算机器中已默认部署了一些 Pod，因此在安装集群前，还要创建至少两台计算机器。

12.7.13. 在 vSphere 中创建更多 Red Hat Enterprise Linux CoreOS（RHCOS）机器

您可以为集群创建更多计算机器，在 VMware vSphere 上使用用户置备的基础架构。

先决条件

- 获取计算机器的 Base64 编码 Ignition 文件。
- 您可以访问您为集群创建的 vSphere 模板。

流程

1. 部署模板后，为集群中的机器部署虚拟机。

a. 右键点击模板的名称，再点击 **Clone → Clone to Virtual Machine**。

b. 在 **Select a name and folder**选项卡中，指定虚拟机的名称。您可以在名称中包含机器类型，如 **compute-1**。

c. 在 **Select a name and folder**选项卡中，选择您为集群创建的文件夹名称。

d. 在 **Select a compute resource** 选项卡中，选择数据中心中的主机名称。

e. 可选：在 **Select storage** 选项卡中，自定义存储选项。

f. 在 **Select clone options** 中，选择 **Customize this virtual machine's hardware**。

g. 在 **Customize hardware** 选项卡中，点击 **VM Options → Advanced**。

- 从 **Latency Sensitivity** 列表中选择 **High**。
- 点击 **Edit Configuration**，然后在 **Configuration Parameters** 窗口中点击 **Add Configuration Params**。定义以下参数名称和值：
  - **guestinfo.ignition.config.data**：粘贴此机器类型的 Base64 编码计算 Ignition 配置文件的内容。
  - **guestinfo.ignition.config.data.encoding**：指定 **base64**。

- **disk.EnableUUID**：指定 **TRUE**。

h. 在 **Customize hardware** 选项卡的 **Virtual Hardware** 面板中，根据需要修改指定的值。确保 RAM、CPU 和磁盘存储的数量满足机器类型的最低要求。另外，如果有多个可用的网络，请确定在 **Add network adapter 中** 选择正确的网络。

i. 完成配置并打开虚拟机电源。

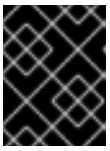
2. 继续为集群创建更多计算机器。

12.7.14. 磁盘分区

在大多数情况下，数据分区最初是由安装 RHCOS 而不是安装另一个操作系统来创建的。在这种情况下，OpenShift Container Platform 安装程序应该被允许配置磁盘分区。

但是，在安装 OpenShift Container Platform 节点时，在两种情况下您可能需要覆盖默认分区：

- 创建单独的分区：对于在空磁盘中的 greenfield 安装，您可能想要在分区中添加单独的存储。这只在生成 **/var** 或者一个 **/var** 独立分区的子目录（如 **/var/lib/etcd** ）时被正式支持，但不支持两者。



重要

Kubernetes 只支持两个文件系统分区。如果您在原始配置中添加多个分区，Kubernetes 无法监控所有这些分区。

- 保留现有分区：对于 brownfield 安装，您要在现有节点上重新安装 OpenShift Container Platform，并希望保留从之前的操作系统中安装的数据分区，对于 **coreos-installer** 来说，引导选项和选项都允许您保留现有数据分区。

创建一个独立的 /var 分区

通常情况下，OpenShift Container Platform 的磁盘分区应该留给安装程序。然而，在有些情况下您可能需要在文件系统的一部分中创建独立分区。

OpenShift Container Platform 支持添加单个分区来将存储附加到 **/var** 分区或 **/var** 的子目录。例如：

- **/var/lib/containers**：保存镜像相关的内容，随着更多镜像和容器添加到系统中，它所占用的存储会增加。
- **/var/lib/etcd**：保存您可能希望保持独立的数据，比如 etcd 存储的性能优化。
- **/var**：保存您希望独立保留的数据，用于特定目的（如审计）。

单独存储 **/var** 目录的内容可方便地根据需要对区域扩展存储，并可以在以后重新安装 OpenShift Container Platform 时保持该数据地完整。使用这个方法，您不必再次拉取所有容器，在更新系统时也无法复制大量日志文件。

因为 **/var** 在进行一个全新的 Red Hat Enterprise Linux CoreOS（RHCOS）安装前必需存在，所以这个流程会在 OpenShift Container Platform 安装过程的 **openshift-install** 准备阶段插入的机器配置来设置独立的 **/var** 分区。

流程

1. 创建存放 OpenShift Container Platform 安装文件的目录：

```
$ mkdir $HOME/clusterconfig
```

2. 运行 **openshift-install** 在 **manifest** 和 **openshift** 子目录中创建一组文件。在出现提示时回答系统问题：

```
$ openshift-install create manifests --dir $HOME/clusterconfig
? SSH Public Key ...
$ ls $HOME/clusterconfig/openshift/
99_kubeadmin-password-secret.yaml
99_openshift-cluster-api_master-machines-0.yaml
99_openshift-cluster-api_master-machines-1.yaml
99_openshift-cluster-api_master-machines-2.yaml
...
```

3. 创建 **MachineConfig** 对象并将其添加到 **openshift** 目录中的一个文件中。例如，把文件命名为 **98-var-partition.yaml**，将磁盘设备名称改为 **worker** 系统中存储设备的名称，并根据情况设置存储大小。这个示例将 **/var** 目录放在一个单独的分区中：

```
apiVersion: machineconfiguration.openshift.io/v1
kind: MachineConfig
metadata:
  labels:
    machineconfiguration.openshift.io/role: worker
  name: 98-var-partition
spec:
  config:
    ignition:
      version: 3.1.0
    storage:
      disks:
      - device: /dev/<device_name> 1
        partitions:
        - label: var
          startMiB: <partition_start_offset> 2
          sizeMiB: <partition_size> 3
      filesystems:
        - device: /dev/disk/by-partlabel/var
          path: /var
          format: xfs
    systemd:
      units:
        - name: var.mount 4
          enabled: true
          contents: |
            [Unit]
            Before=local-fs.target
            [Mount]
            What=/dev/disk/by-partlabel/var
            Where=/var
            Options=defaults,prjquota 5
            [Install]
            WantedBy=local-fs.target
```

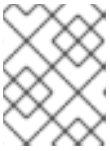
1 要分区的磁盘的存储设备名称。

2 当在引导磁盘中添加数据分区时，推荐最少使用 25000MB。root 文件系统会自动重新定义大小使其占据所有可用空间（最多到指定的偏移值）。如果没有指定值，或者指定的值小于推荐的最小值，则生成的 root 文件系统会太小，而在以后进行的 RHCOS 重新安装可能会覆盖数据分区的开始部分。

3 数据分区的大小（以兆字节为单位）。

4 挂载单元的名称必须与 **Where=** 指令中指定的目录匹配。例如，对于挂载在 **/var/lib/containers** 上的文件系统，该单元必须命名为 **var-lib-containers.mount**。

5 对于用于容器存储的文件系统，必须启用 **prjquota** 挂载选项。



注意

在创建独立 **/var** 分区时，如果不同的实例类型没有相同的设备名称，则无法将不同的实例类型用于 worker 节点。

4. 再次运行 **openshift-install**，从 **manifest** 和 **openshift** 子目录中的一组文件创建 Ignition 配置：

```
$ openshift-install create ignition-configs --dir $HOME/clusterconfig
$ ls $HOME/clusterconfig/
auth  bootstrap.ign  master.ign  metadata.json  worker.ign
```

现在，您可以使用 Ignition 配置文件作为 vSphere 安装程序的输入来安装 Red Hat Enterprise Linux CoreOS（RHCOS）系统。

12.7.15. 创建集群

要创建 OpenShift Container Platform 集群，请等待您通过安装程序生成的 Ignition 配置文件所置备的机器上完成 bootstrap 过程。

先决条件

- 为集群创建所需的基础架构。
- 已获得安装程序并为集群生成了 Ignition 配置文件。
- 已使用 Ignition 配置文件为集群创建 RHCOS 机器。

流程

1. 监控 bootstrap 过程：

```
$ ./openshift-install --dir <installation_directory> wait-for bootstrap-complete \ 1
    --log-level=info 2
```

1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

2 要查看不同的安装详情，请指定 **warn**、**debug** 或 **error**，而不要指定 **info**。

输出示例

```
INFO Waiting up to 30m0s for the Kubernetes API at https://api.test.example.com:6443...
INFO API v1.19.0 up
INFO Waiting up to 30m0s for bootstrapping to complete...
INFO It is now safe to remove the bootstrap resources
```

Kubernetes API 服务器提示已在 control plane 机器上完成 bootstrap 时，命令运行成功。

2. bootstrap 过程完成后，请从负载均衡器中删除 bootstrap 机器。

重要

此时您必须从负载均衡器中删除 bootstrap 机器。您还可以删除或重新格式化机器本身。

12.7.16. 使用 CLI 登录到集群

您可以通过导出集群 **kubeconfig** 文件，以默认系统用户身份登录集群。**kubeconfig** 文件包含关于集群的信息，供 CLI 用于将客户端连接到正确集群和 API 服务器。该文件特只适用于一个特定的集群，在 OpenShift Container Platform 安装过程中创建。

先决条件

- 已部署了 OpenShift Container Platform 集群。
- 已安装 **oc** CLI。

流程

1. 导出 **kubeadmin** 凭证：

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

2. 使用导出的配置，验证能否成功运行 **oc** 命令：

```
$ oc whoami
```

输出示例

```
system:admin
```

12.7.17. 批准机器的证书签名请求

将机器添加到集群时，会为您添加的每台机器生成两个待处理证书签名请求（CSR）。您必须确认这些 CSR 已获得批准，或根据需要自行批准。客户端请求必须首先被批准，然后是服务器请求。

先决条件

- 您已将机器添加到集群中。

流程

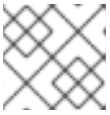
1. 确认集群可以识别这些机器：

```
$ oc get nodes
```

输出示例

```
NAME      STATUS    ROLES   AGE  VERSION
master-0  Ready     master  63m  v1.19.0
master-1  Ready     master  63m  v1.19.0
master-2  Ready     master  64m  v1.19.0
```

输出将列出您创建的所有机器。



注意

在一些 CSR 被批准前，以上输出可能不包括计算节点（也称为 worker 节点）。

2. 检查待处理的 CSR，并确保可以看到添加到集群中的每台机器都有 **Pending** 或 **Approved** 状态的客户端请求：

```
$ oc get csr
```

输出示例

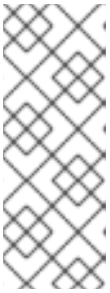
```
NAME        AGE     REQUESTOR                                                   CONDITION
csr-8b2br   15m     system:serviceaccount:openshift-machine-config-operator:node-
bootstrapper   Pending
csr-8vnps   15m     system:serviceaccount:openshift-machine-config-operator:node-
bootstrapper   Pending
...
```

在本例中，两台机器加入了集群。您可能会在列表中看到更多已批准的 CSR。

3. 如果 CSR 没有获得批准，请在所添加机器的所有待处理 CSR 都处于 **Pending** 状态后，为您的集群机器批准这些 CSR：

注意

由于 CSR 会自动轮转，因此请在将机器添加到集群后一小时内批准您的 CSR。如果没有在一小时内批准，证书将会轮转，每个节点将会存在多个证书。您必须批准所有这些证书。批准客户端 CSR 后， Kubelet 为服务证书创建辅助 CSR，这需要手动批准。然后，如果 Kubelet 请求具有相同参数的新证书，则 **machine-approver** 会自动批准后续服务证书续订请求。



注意

对于在未启用机器 API 的平台中运行的集群，如裸机和其他用户置备的基础架构，必须采用一种方法自动批准 kubelet 提供证书请求（CSR）。如果没有批准请求，则 **oc exec**、**oc rsh** 和 **oc logs** 命令将无法成功，因为 API 服务器连接到 kubelet 时需要服务证书。与 Kubelet 端点联系的任何操作都需要此证书批准。这个方法必须监视新的 CSR，确认 CSR 由 **system:node** 或 **system:admin** 组中的 **node-bootstrapper** 服务帐户提交，并确认节点的身份。

- 若要单独批准，请对每个有效的 CSR 运行以下命令：

```
$ oc adm certificate approve <csr_name> 1
```

1 **<csr_name>** 是当前 CSR 列表中 CSR 的名称。

- 要批准所有待处理的 CSR，请运行以下命令：

```
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}{{"\n"}}
{{end}}{{end}}' | xargs --no-run-if-empty oc adm certificate approve
```

注意

在有些 CSR 被批准前，一些 Operator 可能无法使用。

4. 现在，您的客户端请求已被批准，您必须查看添加到集群中的每台机器的服务器请求：

```
$ oc get csr
```

输出示例

```
NAME        AGE     REQUESTOR                                                   CONDITION
csr-bfd72   5m26s   system:node:ip-10-0-50-126.us-east-2.compute.internal
Pending
csr-c57lv   5m26s   system:node:ip-10-0-95-157.us-east-2.compute.internal
Pending
...
```

5. 如果剩余的 CSR 没有被批准，且处于 **Pending** 状态，请批准集群机器的 CSR：

- 若要单独批准，请对每个有效的 CSR 运行以下命令：

```
$ oc adm certificate approve <csr_name> 1
```

1 **<csr_name>** 是当前 CSR 列表中 CSR 的名称。

- 要批准所有待处理的 CSR，请运行以下命令：

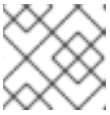
```
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}{{"\n"}}
{{end}}{{end}}' | xargs oc adm certificate approve
```

6. 批准所有客户端和服务器 CSR 后，器将处于 **Ready** 状态。运行以下命令验证：

```
$ oc get nodes
```

输出示例

```
NAME     STATUS   ROLES   AGE  VERSION
master-0 Ready    master  73m  v1.20.0
master-1 Ready    master  73m  v1.20.0
master-2 Ready    master  74m  v1.20.0
worker-0 Ready    worker  11m  v1.20.0
worker-1 Ready    worker  11m  v1.20.0
```



注意

批准服务器 CSR 后可能需要几分钟时间让机器转换为 **Ready** 状态。

其他信息

- 如需有关 CSR 的更多信息，请参阅证书签名请求。

12.7.18. 初始 Operator 配置

在 control plane 初始化后，您必须立即配置一些 Operator 以便它们都可用。

先决条件

- 您的 control plane 已初始化。

流程

1. 观察集群组件上线：

```
$ watch -n5 oc get clusteroperators
```

输出示例

```
NAME                                       VERSION AVAILABLE   PROGRESSING   DEGRADED
SINCE
authentication                             4.6.0   True        False         False      3h56m
cloud-credential                           4.6.0   True        False         False      29h
cluster-autoscaler                         4.6.0   True        False         False      29h
config-operator                            4.6.0   True        False         False      6h39m
console                                    4.6.0   True        False         False      3h59m
csi-snapshot-controller                    4.6.0   True        False         False      4h12m
dns                                        4.6.0   True        False         False      4h15m
etcd                                       4.6.0   True        False         False      29h
image-registry                             4.6.0   True        False         False      3h59m
ingress                                    4.6.0   True        False         False      4h30m
insights                                   4.6.0   True        False         False      29h
kube-apiserver                             4.6.0   True        False         False      29h
kube-controller-manager                    4.6.0   True        False         False      29h
kube-scheduler                             4.6.0   True        False         False      29h
kube-storage-version-migrator              4.6.0   True        False         False      4h2m
machine-api                                4.6.0   True        False         False      29h
```

```
machine-approver                           4.6.0   True        False        False      6h34m
machine-config                             4.6.0   True        False        False      3h56m
marketplace                                4.6.0   True        False        False      4h2m
monitoring                                 4.6.0   True        False        False      6h31m
network                                    4.6.0   True        False        False      29h
node-tuning                                4.6.0   True        False        False      4h30m
openshift-apiserver                        4.6.0   True        False        False      3h56m
openshift-controller-manager               4.6.0   True        False        False      4h36m
openshift-samples                          4.6.0   True        False        False      4h30m
operator-lifecycle-manager                 4.6.0   True        False        False      29h
operator-lifecycle-manager-catalog         4.6.0   True        False        False      29h
operator-lifecycle-manager-packageserver   4.6.0   True        False        False      3h59m
service-ca                                 4.6.0   True        False        False      29h
storage                                    4.6.0   True        False        False      4h30m
```

2. 配置不可用的 Operator。

12.7.18.1. 禁用默认的 OperatorHub 源

在 OpenShift Container Platform 安装过程中，默认为 OperatorHub 配置由红帽和社区项目提供的源内容的 operator 目录。在受限网络环境中，必须以集群管理员身份禁用默认目录。

流程

- 通过在 **OperatorHub** 对象中添加 **disableAllDefaultSources: true** 来禁用默认目录的源：

```
$ oc patch OperatorHub cluster --type json \
    -p '[{"op": "add", "path": "/spec/disableAllDefaultSources", "value": true}]'
```

提示

或者，您可以使用 Web 控制台管理目录源。在 **Administration → Cluster Settings → Global Configuration → OperatorHub** 页面中，点 **Sources** 选项卡，其中可创建、删除、禁用和启用单独的源。

12.7.18.2. 镜像 registry 存储配置

对于不提供默认存储的平台，Image Registry Operator 最初将不可用。安装后，您必须配置 registry 使用的存储，这样 Registry Operator 才可用。

示配置生产集群所需的持久性卷的说明。如果适用，显示有关将空目录配置为存储位置的说明，该位置只可用于非生产集群。

另外还提供了在升级过程中使用 **Recreate** rollout 策略来允许镜像 registry 使用块存储类型的说明。

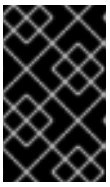
12.7.18.2.1. 为 VMware vSphere 配置 registry 存储

作为集群管理员，在安装后需要配置 registry 来使用存储。

先决条件

- 具有 Cluster Administrator 权限
- VMware vSphere上有一个集群。

- 为集群置备的持久性存储，如 Red Hat OpenShift Container Storage。



重要

如果您只有一个副本，OpenShift Container Platform 支持对镜像 registry 存储的 **ReadWriteOnce** 访问。要部署支持高可用性的、带有两个或多个副本的镜像 registry，需要 **ReadWriteMany** 访问设置。

- 必须有“ 100Gi”容量。

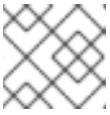
重要

测试显示，在 RHEL 中使用 NFS 服务器作为核心服务的存储后端可能会出现问题。这包括 OpenShift Container Registry 和 Quay，Prometheus 用于监控存储，以及 Elasticsearch 用于日志存储。因此，不推荐使用 RHEL NFS 作为 PV 后端用于核心服务。

市场上的其他 NFS 实现可能没有这些问题。如需了解更多与此问题相关的信息，请联络相关的 NFS 厂商。

流程

1. 为了配置 registry 使用存储，需要修改 **configs.imageregistry/cluster** 资源中的 **spec.storage.pvc** 。



注意

使用共享存储时，请查看您的安全设置以防止被外部访问。

2. 验证您没有 registry pod:

```
$ oc get pod -n openshift-image-registry
```

注意

如果存储类型为 **emptyDIR**，则副本数不能超过 **1**。

3. 检查 registry 配置：

```
$ oc edit configs.imageregistry.operator.openshift.io
```

输出示例

```
storage:
  pvc:
    claim: 1
```

1 将 **claim** 字段留空以允许自动创建一个 **image-registry-storage** PVC。

4. 检查 **clusteroperator** 的状态：

```
$ oc get clusteroperator image-registry
```

12.7.18.2.2. 在非生产集群中配置镜像 registry 存储

您必须为 Image Registry Operator 配置存储。对于非生产集群，您可以将镜像 registry 设置为空目录。如果您这样做，重启 registry 后会丢失所有镜像。

流程

- 将镜像 registry 存储设置为空目录：

```
$ oc patch configs.imageregistry.operator.openshift.io cluster --type merge --patch '{"spec":{"storage":{"emptyDir":{}}}}'
```

> **警告**
>
> 仅可为非生产集群配置这个选项。

如果在 Image Registry Operator 初始化其组件前运行此命令，**oc patch** 命令会失败并显示以下错误：

```
Error from server (NotFound): configs.imageregistry.operator.openshift.io "cluster" not found
```

等待几分钟，然后再次运行该命令。

12.7.18.2.3. 为 VMware vSphere 配置块 registry 存储

在作为集群管理员升级时，要允许镜像 registry 使用块存储类型，如 vSphere Virtual Machine Disk（VMDK），您可以使用 **Recreate** rollout 策略。

> **重要**
>
> 支持块存储卷，但不建议将其用于生产环境中的镜像 registry。在块存储上配置 registry 的安装不具有高可用性，因为 registry 无法拥有多个副本。

流程

1. 要将镜像 registry 存储设置为块存储类型，对 registry 进行补丁，使其使用 **Recreate** rollout 策略，且仅使用 **1** 个副本运行：

```
$ oc patch config.imageregistry.operator.openshift.io/cluster --type=merge -p '{"spec":{"rolloutStrategy":"Recreate","replicas":1}}'
```

2. 为块存储设备置备 PV，并为该卷创建 PVC。请求的块卷使用 ReadWriteOnce（RWO）访问模式。

   a. 创建包含以下内容的 **pvc.yaml** 文件以定义 VMware vSphere **PersistentVolumeClaim** ：

```
kind: PersistentVolumeClaim
```

```
apiVersion: v1
metadata:
  name: image-registry-storage 1
  namespace: openshift-image-registry 2
spec:
  accessModes:
  - ReadWriteOnce 3
  resources:
    requests:
      storage: 100Gi 4
```

1. 代表 **PersistentVolumeClaim** 对象的唯一名称。
2. **PersistentVolumeClaim** 对象的命名空间，即 **openshift-image-registry**。
3. 持久性卷声明的访问模式。使用 **ReadWriteOnce** 时，单个节点可以通过读写权限挂载这个卷。
4. 持久性卷声明的大小。

b. 从文件创建 **PersistentVolumeClaim** 对象：

```
$ oc create -f pvc.yaml -n openshift-image-registry
```

3. 编辑 registry 配置，使其可以正确引用 PVC:

```
$ oc edit config.imageregistry.operator.openshift.io -o yaml
```

输出示例

```
storage:
  pvc:
    claim: 1
```

1. 通过创建自定义 PVC，您可以将 **claim** 字段留空以用于默认自动创建 **image-registry-storage** PVC。

有关配置 registry 存储以便引用正确的 PVC 的说明，请参阅为 VMware vSphere 配置 registry 存储。

12.7.19. 在用户置备的基础架构上完成安装

完成 Operator 配置后，可以在您提供的基础架构上完成集群安装。

先决条件

- 您的 control plane 已初始化。
- 已完成初始 Operator 配置。

流程

1. 使用以下命令确认所有集群组件都已在线：

```
$ watch -n5 oc get clusteroperators
```

输出示例

```
NAME                                       VERSION AVAILABLE   PROGRESSING   DEGRADED
SINCE
authentication                             4.6.0   True        False         False      3h56m
cloud-credential                           4.6.0   True        False         False      29h
cluster-autoscaler                         4.6.0   True        False         False      29h
config-operator                            4.6.0   True        False         False      6h39m
console                                    4.6.0   True        False         False      3h59m
csi-snapshot-controller                    4.6.0   True        False         False      4h12m
dns                                        4.6.0   True        False         False      4h15m
etcd                                       4.6.0   True        False         False      29h
image-registry                             4.6.0   True        False         False      3h59m
ingress                                    4.6.0   True        False         False      4h30m
insights                                   4.6.0   True        False         False      29h
kube-apiserver                             4.6.0   True        False         False      29h
kube-controller-manager                    4.6.0   True        False         False      29h
kube-scheduler                             4.6.0   True        False         False      29h
kube-storage-version-migrator              4.6.0   True        False         False      4h2m
machine-api                                4.6.0   True        False         False      29h
machine-approver                           4.6.0   True        False         False      6h34m
machine-config                             4.6.0   True        False         False      3h56m
marketplace                                4.6.0   True        False         False      4h2m
monitoring                                 4.6.0   True        False         False      6h31m
network                                    4.6.0   True        False         False      29h
node-tuning                                4.6.0   True        False         False      4h30m
openshift-apiserver                        4.6.0   True        False         False      3h56m
openshift-controller-manager               4.6.0   True        False         False      4h36m
openshift-samples                          4.6.0   True        False         False      4h30m
operator-lifecycle-manager                 4.6.0   True        False         False      29h
operator-lifecycle-manager-catalog         4.6.0   True        False         False      29h
operator-lifecycle-manager-packageserver   4.6.0   True        False         False      3h59m
service-ca                                 4.6.0   True        False         False      29h
storage                                    4.6.0   True        False         False      4h30m
```

或者，通过以下命令，如果所有集群都可用您会接到通知。它还检索并显示凭证：

```
$ ./openshift-install --dir <installation_directory> wait-for install-complete 1
```

1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

输出示例

```
INFO Waiting up to 30m0s for the cluster to initialize...
```

Cluster Version Operator 完成从 Kubernetes API 服务器部署 OpenShift Container Platform 集群时，命令运行成功。

重要

- 安装程序生成的 Ignition 配置文件包含在 24 小时后过期的证书，然后在过期时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外情况是，您需要手动批准待处理的 **node-bootstrapper** 证书签名请求（CSR）来恢复 kubelet 证书。如需更多信息，请参阅*从过期的 control plane 证书中恢复*的文档。
- 建议您在生成 12 小时后使用 Ignition 配置文件，因为集群安装后 24 小时证书从 16 小时轮转至 22 小时。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中运行证书更新时避免安装失败。

2. 确认 Kubernetes API 服务器正在与 pod 通信。

   a. 要查看所有 pod 的列表，请使用以下命令：

   ```
   $ oc get pods --all-namespaces
   ```

   输出示例

   ```
   NAMESPACE                       NAME                                         READY   STATUS
   RESTARTS   AGE
   openshift-apiserver-operator    openshift-apiserver-operator-85cb746d55-zqhs8   1/1
   Running   1          9m
   openshift-apiserver             apiserver-67b9g                               1/1     Running   0
   3m
   openshift-apiserver             apiserver-ljcmx                               1/1     Running   0
   1m
   openshift-apiserver             apiserver-z25h4                               1/1     Running   0
   2m
   openshift-authentication-operator authentication-operator-69d5d8bf84-vh2n8     1/1
   Running   0          5m
   ...
   ```

   b. 使用以下命令，查看上一命令的输出中所列 pod 的日志：

   ```
   $ oc logs <pod_name> -n <namespace>
   ```
   (1)

   (1) 指定 pod 名称和命名空间，如上一命令的输出中所示。

   如果 pod 日志显示，Kubernetes API 服务器可以与集群机器通信。

3. 在 Cluster registration 页面注册您的集群。

您可以按照将计算机器添加到 vSphere 的内容，在集群安装完成后添加额外的计算机器。

12.7.20. 备份 VMware vSphere 卷

OpenShift Container Platform 将新卷作为独立持久性磁盘置备，以便在集群中的任何节点上自由附加和分离卷。因此，无法备份使用快照的卷，也无法从快照中恢复卷。如需更多信息，请参阅 快照限制。

流程

要创建持久性卷的备份：

1. 停止使用持久性卷的应用程序。
2. 克隆持久性卷。
3. 重启应用程序。
4. 创建克隆的卷的备份。
5. 删除克隆的卷。

12.7.21. OpenShift Container Platform 的 Telemetry 访问

在 OpenShift Container Platform 4.6 中，默认运行的 Telemetry 服务提供有关集群健康状况和成功更新的指标，需要访问互联网。如果您的集群连接到互联网，Telemetry 会自动运行，而且集群会注册到 OpenShift Cluster Manager。

确认 OpenShift Cluster Manager 清单正确后，可以由 Telemetry 自动维护，也可以使用 OpenShift Cluster Manager 手动维护，使用订阅监控来跟踪帐户或多集群级别的 OpenShift Container Platform 订阅。

其他资源

- 有关 Telemetry 服务的更多信息，请参阅关于远程健康监控。

12.7.22. 后续步骤

- 自定义集群。
- 为 Cluster Samples Operator 和 **must-gather** 工具配置镜像流。
- 了解如何在受限网络中使用 Operator Lifecycle Manager（OLM）。
- 如果您用来安装集群的镜像 registry 具有一个可信任的 CA，通过配置额外的信任存储将其添加到集群中。
- 如果需要，您可以选择不使用远程健康报告。

12.8. 在 VMC 上卸载集群

您可以删除使用安全程序置备的基础架构部署到 VMware Cloud (VMC) on AWS 中的基础架构上安装的集群。

12.8.1. 删除使用安装程序置备的基础架构的集群

您可以从云中删除使用安装程序置备的基础架构的集群。

注意

卸载后，检查云供应商是否有没有被正确移除的资源，特别是 User Provisioned Infrastructure（UPI）集群。可能存在安装程序没有创建的资源，或者安装程序无法访问的资源。

先决条件

- 有部署集群时所用的安装程序副本。
- 有创建集群时安装程序所生成的文件。

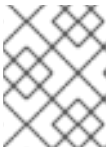
流程

1. 在用来安装集群的计算机中包含安装程序的目录中，运行以下命令：

```
$ ./openshift-install destroy cluster \
--dir <installation_directory> --log-level info 1 2
```

1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

2 要查看不同的详情，请指定 **warn**、**debug** 或 **error**，而不要指定 **info**。



注意

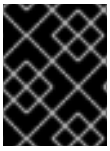
您必须为集群指定包含集群定义文件的目录。安装程序需要此目录中的 **metadata.json** 文件来删除集群。

2. 可选：删除 **<installation_directory>** 目录和 OpenShift Container Platform 安装程序。

第 13 章 在任意平台上安装

13.1. 在任何平台上安装集群

在 OpenShift Container Platform 版本 4.6 中，您可以在您置备的任何基础架构上安装集群，包括虚拟化和云环境。

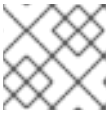


重要

在尝试在虚拟化或云环境中安装 OpenShift Container Platform 集群前，请参阅有关在未经测试的平台上部署 OpenShift Container Platform 的指南中的信息。

13.1.1. 先决条件

- 查看有关 OpenShift Container Platform 安装和更新流程的详细信息。
- 如果使用防火墙，则必须将其配置为允许集群需要访问的站点。



注意

如果您要配置代理，请务必也要查看此站点列表。

13.1.2. OpenShift Container Platform 的互联网访问

在 OpenShift Container Platform 4.6 中，您需要访问互联网来安装集群。

您必须具有以下互联网访问权限：

- 访问 OpenShift Cluster Manager 以下载安装程序并执行订阅管理。如果集群可以访问互联网，并且没有禁用 Telemetry，该服务会自动授权您的集群。
- 访问 Quay.io，以获取安装集群所需的软件包。
- 获取执行集群更新所需的软件包。

重要

如果您的集群无法直接访问互联网，则可以在置备的某些类基础架构上执行受限网络安装。在此过程中，您要下载所需的内容，并使用它在镜像 registry（mirror registry） 中填充安装集群并生成安装程序所需的软件包。对于某些安装类型，集群要安装到的环境不需要访问互联网。在更新集群之前，要更新 registry 镜像系统中的内容。

13.1.3. 具有用户置备基础架构的集群的机器要求

对于含有用户置备的基础架构的集群，您必须部署所有所需的机器。

13.1.3.1. 所需的机器

最小的 OpenShift Container Platform 集群需要下列主机：

- 一个临时 bootstrap 机器
- 三台 control plane 或 master 机器

- 至少两台计算机器，也称为 worker 机器。

注意

集群要求 bootstrap 机器在三台 control plane 机器上部署 OpenShift Container Platform 集群。您可在安装集群后删除 bootstrap 机器。

重要

要保持集群的高可用性，请将独立的物理主机用于这些集群机器。

bootstrap 和 control plane 机器必须使用 Red Hat Enterprise Linux CoreOS (RHCOS) 作为操作系统。但是，计算机器可以在 Red Hat Enterprise Linux CoreOS(RHCOS)或 Red Hat Enterprise Linux(RHEL)7.9 之间进行选择。

请注意，RHCOS 基于 Red Hat Enterprise Linux（RHEL） 8，并继承其所有硬件认证和要求。请查看Red Hat Enterprise Linux 技术功能及限制 。

13.1.3.2. 网络连接要求

所有 Red Hat Enterprise Linux CoreOS (RHCOS) 机器在启动过程中需要 **initramfs** 中的网络从 Machine Config Server 获取 Ignition 配置文件。在初次启动过程中，需要一个 DHCP 服务器或设置了静态 IP 地址来建立网络连接，以下载它们的 Ignition 配置文件。另外，集群中的每个 OpenShift Container Platform 节点都必须有权访问网络时间协议（NTP）服务器。如果 DHCP 服务器提供 NTP 服务器信息，Red Hat Enterprise Linux CoreOS（RHCOS）机器上的 chrony 时间服务会读取信息，并可与 NTP 服务器同步时钟。

13.1.3.3. 最低资源要求

每台集群机器都必须满足以下最低要求：

表 13.1. 最低资源要求

机器	操作系统	vCPU [1]	虚拟内存	存储	IOPS [2]
bootstrap	RHCOS	4	16 GB	100 GB	300
Control plane	RHCOS	4	16 GB	100 GB	300
Compute	RHCOS 或 RHEL 7.9	2	8 GB	100 GB	300

1. 当未启用并发多线程 (SMT) 或超线程时，一个 vCPU 相当于一个物理内核。启用后，使用以下公式来计算对应的比例：（每个内核数的线程）× sockets = vCPU。

2. OpenShift Container Platform 和 Kubernetes 对磁盘性能非常敏感，建议使用更快的存储速度，特别是 control plane 节点上需要 10 ms p99 fsync 持续时间的 etcd。请注意，在许多云平台上，存储大小和 IOPS 可一起扩展，因此您可能需要过度分配存储卷来获取足够的性能。

13.1.3.4. 证书签名请求管理

在使用您置备的基础架构时，集群只能有限地访问自动机器管理，因此您必须提供一种在安装后批准集群

证书签名请求 (CSR) 的机制。**kube-controller-manager** 只能批准 kubelet 客户端 CSR。 **machine-approver** 无法保证使用 kubelet 凭证请求的提供证书的有效性，因为它不能确认是正确的机器发出了该请求。您必须决定并实施一种方法，以验证 kubelet 提供证书请求的有效性并进行批准。

13.1.4. 创建用户置备的基础架构

在部署采用用户置备的基础架构的 OpenShift Container Platform 集群前，您必须创建底层基础架构。

先决条件

- 在为集群创建支持基础架构之前，请参阅OpenShift Container Platform 4.x Tested Integrations页。

流程

1. 在每个节点上配置 DHCP 或设置静态 IP 地址。
2. 提供所需的负载均衡器。
3. 配置机器的端口。
4. 配置 DNS。
5. 确保网络可以正常工作。

13.1.4.1. 用户置备的基础架构对网络的要求

所有 Red Hat Enterprise Linux CoreOS（RHCOS）机器在启动过程中需要 **initramfs** 中的网络从机器配置服务器获取 Ignition 配置。

在初次启动过程中，需要一个 DHCP 服务器或集群中的每个机器都设置了静态 IP 地址来建立网络连接，以下载它们的 Ignition 配置文件。

建议您使用 DHCP 服务器为集群进行长期机器管理。确保 DHCP 服务器已配置为向集群机器提供持久 IP 地址和主机名。

Kubernetes API 服务器必须能够解析集群机器的节点名称。如果 API 服务器和 worker 节点位于不同的区域中，您可以配置默认 DNS 搜索区域，以便 API 服务器能够解析节点名称。另一种支持的方法是始终在节点对象和所有 DNS 请求中使用完全限定域名来指代主机。

您必须配置机器间的网络连接，以便集群组件进行通信。每台机器都必须能够解析集群中所有其他机器的主机名。

表 13.2. 所有机器到所有机器

协议	端口	描述
ICMP	N/A	网络可访问性测试
TCP	**1936**	指标
	9000-**9999**	主机级别的服务，包括端口 **9100**-**9101** 上的节点导出器和端口 **9099** 上的 Cluster Version Operator。

协议	端口	描述
	10250-10259	Kubernetes 保留的默认端口
	10256	openshift-sdn
UDP	**4789**	VXLAN 和 Geneve
	6081	VXLAN 和 Geneve
	9000-9999	主机级别的服务，包括端口 **9100-9101** 上的节点导出器。
TCP/UDP	**30000-32767**	Kubernetes 节点端口

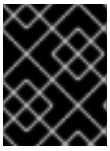
表 13.3. 要通过控制平面的所有机器

协议	端口	描述
TCP	**6443**	Kubernetes API

表 13.4. control plane 机器到 control plane 机器

协议	端口	描述
TCP	**2379-2380**	etcd 服务器和对等端口

网络拓扑要求
您为集群置备的基础架构必须满足下列网络拓扑要求。



重要

OpenShift Container Platform 要求所有节点都能访问互联网，以便为平台容器提取镜像并向红帽提供遥测数据。

负载均衡器
在安装 OpenShift Container Platform 前，您必须置备两个满足以下要求的负载均衡器：

1. **API 负载均衡器**：提供一个通用端点，供用户（包括人和机器）与平台交互和配置。配置以下条件：
   - 只适用于第 4 层负载均衡。这可被称为 Raw TCP、SSL Passthrough 或者 SSL 桥接模式。如果使用 SSL Bridge 模式，必须为 API 路由启用 Server Name Indication（SNI）。
   - 无状态负载平衡算法。这些选项根据负载均衡器的实现而有所不同。

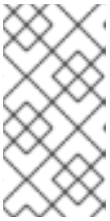
重要

不要为 API 负载均衡器配置会话持久性。

在负载均衡器的前端和后台配置以下端口：

表 13.5. API 负载均衡器

端口	后端机器（池成员）	内部	外部	描述
6443	Bootstrap 和 control plane.bootstrap 机器初始化集群 control plane 后，您要从负载均衡器中删除 bootstrap 机器。您必须为 API 服务器健康检查探测配置 **/readyz** 端点。	X	X	Kubernetes API 服务器
22623	Bootstrap 和 control plane.bootstrap 机器初始化集群 control plane 后，您要从负载均衡器中删除 bootstrap 机器。	X		机器配置服务器



注意

负载均衡器必须配置为，从 API 服务器关闭 **/readyz** 端点到从池中删除 API 服务器实例时最多需要 30 秒。在 **/readyz** 返回错误或处于健康状态后的时间范围内，端点必须被删除或添加。每 5 秒或 10 秒探测一次，有两个成功请求处于健康状态，三个成为不健康的请求经过测试。

2. **应用程序入口负载均衡器**:提供来自集群外部的应用程序流量流量的 Ingress 点。配置以下条件：

   - 只适用于第 4 层负载均衡。这可被称为 Raw TCP、SSL Passthrough 或者 SSL 桥接模式。如果使用 SSL Bridge 模式，您必须为 Ingress 路由启用Server Name Indication（SNI）。
   - 建议根据可用选项以及平台上托管的应用程序类型，使用基于连接的或者基于会话的持久性。

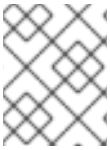
   在负载均衡器的前端和后台配置以下端口：

   表 13.6. 应用程序入口负载均衡器

端口	后端机器（池成员）	内部	外部	描述
443	默认运行入口路由器 Pod、计算或 worker 的机器。	X	X	HTTPS 流量
80	默认运行入口路由器 Pod、计算或 worker 的机器。	X	X	HTTP 流量

提示

如果负载均衡器可以看到客户端的真实 IP 地址，启用基于 IP 的会话持久性可提高使用端到端 TLS 加密的应用程序的性能。



注意

OpenShift Container Platform 集群需要正确配置入口路由器。control plane 初始化后，您必须配置入口路由器。

NTP 配置

OpenShift Container Platform 集群默认配置为使用公共网络时间协议（NTP）服务器。如果要使用本地企业 NTP 服务器，或者集群部署在断开连接的网络中，您可以将集群配置为使用特定的时间服务器。如需更多信息，请参阅*配置 chrony 时间服务*的文档。

如果 DHCP 服务器提供 NTP 服务器信息，Red Hat Enterprise Linux CoreOS（RHCOS）机器上的 chrony 时间服务会读取信息，并可与 NTP 服务器同步时钟。

其他资源

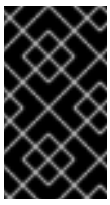
- 配置 chrony 时间服务

13.1.4.2. 用户置备 DNS 要求

DNS 用于名称解析和反向名称解析。DNS A/AAAA 或 CNAME 记录用于名称解析，PTR 记录用于反向解析名称。反向记录很重要，因为 Red Hat Enterprise Linux CoreOS（RHCOS）使用反向记录为所有节点设置主机名。另外，反向记录用于生成 OpenShift Container Platform 需要操作的证书签名请求（CSR）。

采用用户置备的基础架构的 OpenShift Container Platform 集群需要以下 DNS 记录。在每一记录中，**<cluster_name>** 是集群名称，**<base_domain>** 则是您在 **install-config.yaml** 文件中指定的集群基域。完整的 DNS 记录采用如下格式: **<component>.<cluster_name>.<base_domain>.**。

表 13.7. 所需的 DNS 记录

组件	记录	描述
Kubernetes API	**api.<cluster_name>.<base_domain>.**	添加 DNS A/AAAA 或 CNAME 记录，以及 DNS PTR 记录，以识别 control plane 机器的负载均衡器。这些记录必须由集群外的客户端以及集群中的所有节点解析。
	api-int.<cluster_name>.<base_domain>.	添加 DNS A/AAAA 或 CNAME 记录，以及 DNS PTR 记录，以识别 control plane 机器的负载均衡器。这些记录必须可以从集群中的所有节点解析。  **重要** API 服务器必须能够根据在 Kubernetes 中记录的主机名解析 worker 节点。如果 API 服务器无法解析节点名称，则代理的 API 调用会失败，且您无法从 pod 检索日志。
Routes	***.apps.<cluster_name>.<base_domain>.**	添加通配符 DNS A/AAAA 或 CNAME 记录，指向以运行入口路由器 Pod 的机器（默认为 worker 节点）为目标的负载均衡器。这些记录必须由集群外的客户端以及集群中的所有节点解析。

组件	记录	描述
bootstrap	**bootstrap.<cluster_name>.<base_domain>.**	添加 DNS A/AAAA 或 CNAME 记录，以及 DNS PTR 记录来识别 bootstrap 机器。这些记录必须由集群中的节点解析。
Master 主机	**<master><n>.<cluster_name>.<base_domain>.**	DNS A/AAAA 或 CNAME 记录，以识别 control plane 节点（也称为 master 节点）的每台机器。这些记录必须由集群中的节点解析。
Worker 主机	**<worker><n>.<cluster_name>.<base_domain>.**	添加 DNS A/AAAA 或 CNAME 记录，以识别 worker 节点的每台机器。这些记录必须由集群中的节点解析。

提示

您可以使用 **nslookup <hostname>** 命令来验证名称解析。您可以使用 **dig -x <ip_address>** 命令来验证 PTR 记录的反向名称解析。

下面的 BIND 区文件的例子展示了关于名字解析的 A 记录的例子。这个示例的目的是显示所需的记录。这个示例不是为选择一个名称解析服务提供建议。

例 13.1. DNS 区数据库示例

```
$TTL 1W
@ IN SOA ns1.example.com. root (
   2019070700 ; serial
   3H  ; refresh (3 hours)
   30M  ; retry (30 minutes)
   2W  ; expiry (2 weeks)
   1W )  ; minimum (1 week)
 IN NS ns1.example.com.
 IN MX 10 smtp.example.com.
;
;
ns1 IN A 192.168.1.5
smtp IN A 192.168.1.5
;
helper IN A 192.168.1.5
helper.ocp4 IN A 192.168.1.5
;
; The api identifies the IP of your load balancer.
api.ocp4  IN A 192.168.1.5
api-int.ocp4  IN A 192.168.1.5
;
; The wildcard also identifies the load balancer.
*.apps.ocp4  IN A 192.168.1.5
;
; Create an entry for the bootstrap host.
bootstrap.ocp4 IN A 192.168.1.96
;
; Create entries for the master hosts.
master0.ocp4  IN A 192.168.1.97
```

```
master1.ocp4  IN A 192.168.1.98
master2.ocp4  IN A 192.168.1.99
;
; Create entries for the worker hosts.
worker0.ocp4  IN A 192.168.1.11
worker1.ocp4  IN A 192.168.1.7
;
;EOF
```

下面的 BIND 区文件示例显示了反向名字解析的 PTR 记录示例。

例 13.2. 反向记录的 DNS 区数据库示例

```
$TTL 1W
@ IN SOA ns1.example.com. root (
   2019070700 ; serial
   3H  ; refresh (3 hours)
   30M  ; retry (30 minutes)
   2W  ; expiry (2 weeks)
   1W )  ; minimum (1 week)
 IN NS ns1.example.com.
;
; The syntax is "last octet" and the host must have an FQDN
; with a trailing dot.
97 IN PTR master0.ocp4.example.com.
98 IN PTR master1.ocp4.example.com.
99 IN PTR master2.ocp4.example.com.
;
96 IN PTR bootstrap.ocp4.example.com.
;
5 IN PTR api.ocp4.example.com.
5 IN PTR api-int.ocp4.example.com.
;
11 IN PTR worker0.ocp4.example.com.
7 IN PTR worker1.ocp4.example.com.
;
;EOF
```

13.1.5. 生成 SSH 私钥并将其添加到代理中

如果要在集群上执行安装调试或灾难恢复，则必须为 **ssh-agent** 和安装程序提供 SSH 密钥。您可以使用此密钥访问公共集群中的 bootstrap 机器来排除安装问题。

注意

在生产环境中，您需要进行灾难恢复和调试。

您可以使用此密钥以 **core** 用户身份通过 SSH 连接到 master 节点。在部署集群时，此密钥会添加到 **core** 用户的 **~/.ssh/authorized_keys** 列表中。

注意

您必须使用一个本地密钥，而不要使用在特定平台上配置的密钥，如 AWS 密钥对。

流程

1. 如果还没有为计算机上免密码身份验证而配置的 SSH 密钥，请创建一个。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ ssh-keygen -t ed25519 -N '' \
  -f <path>/<file_name> 1
```

1 指定新 SSH 密钥的路径和文件名，如 **~/.ssh/id_rsa**。如果您已有密钥对，请确保您的公钥位于 **~/.ssh** 目录中。

运行此命令会在指定的位置生成不需要密码的 SSH 密钥。

注意

如果您计划在 **x86_64** 架构中安装使用 FIPS 验证的/Modules in Process 加密库的 OpenShift Container Platform 集群，不要创建使用 **ed25519** 算法的密钥。反之，创建一个使用 **rsa** 或 **ecdsa** 算法的密钥。

2. 作为后台任务启动 **ssh-agent** 进程：

```
$ eval "$(ssh-agent -s)"
```

输出示例

```
Agent pid 31874
```

注意

如果您的集群采用 FIPS 模式，则只使用 FIPS 兼容算法来生成 SSH 密钥。密钥必须是 RSA 或 ECDSA。

3. 将 SSH 私钥添加到 **ssh-agent**：

```
$ ssh-add <path>/<file_name> 1
```

输出示例

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

1 指定 SSH 私钥的路径和文件名，如 **~/.ssh/id_rsa**

后续步骤

- 在安装 OpenShift Container Platform 时，为安装程序提供 SSH 公钥。如果在您置备的基础架构上安装集群，您必须将此密钥提供给集群的机器。

13.1.6. 获取安装程序

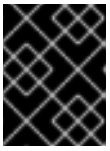
在安装 OpenShift Container Platform 之前，将安装文件下载到本地计算机上。

先决条件

- 运行 Linux 或 macOS 的计算机，本地磁盘空间为 500 MB

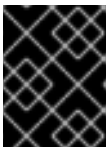
流程

1. 访问 OpenShift Cluster Manager 站点的 Infrastructure Provider 页面。如果您有红帽帐号，请使用自己的凭证登录。如果没有，请创建一个帐户。
2. 选择您的基础架构供应商。
3. 进入适用于您的安装类型的页面，下载您的操作系统的安装程序，并将文件放在要保存安装配置文件的目录中。。



重要

安装程序会在用来安装集群的计算机上创建若干文件。在完成集群安装后，您必须保留安装程序和安装程序所创建的文件。这两个文件都需要删除集群。



重要

删除安装程序创建的文件不会删除您的集群，即使集群在安装过程中失败也是如此。要删除集群，为特定云供应商完成 OpenShift Container Platform 卸载流程。

4. 提取安装程序。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ tar xvf openshift-install-linux.tar.gz
```

5. 从 Red Hat OpenShift Cluster Manager 下载安装 pull secret 。通过此 pull secret，您可以进行所含授权机构提供的服务的身份验证，这些服务包括为 OpenShift Container Platform 组件提供容器镜像的 Quay.io。

13.1.7. 通过下载二进制文件安装 OpenShift CLI

您需要安装 CLI（**oc**） 来使用命令行界面与 OpenShift Container Platform 进行交互。您可在 Linux 、Windows 或 macOS 上安装 **oc**。

重要

如果安装了旧版本的 **oc**，则无法使用 OpenShift Container Platform 4.6 中的所有命令。下载并安装新版本的 **oc**。

13.1.7.1. 在 Linux 上安装 OpenShift CLI

您可以按照以下流程在 Linux 上安装 OpenShift CLI（**oc**）二进制文件。

流程

1. 进入到红帽客户门户网站上的 OpenShift Container Platform 下载页面。

2. 在 **Version** 下拉菜单中选择相应的版本。
3. 单击 **OpenShift v4.6 Linux 客户端** 条目旁边的 **Download Now**，再保存文件。
4. 解包存档：

```
$ tar xvzf <file>
```

5. 把 **oc** 二进制代码放到 **PATH** 中的目录中。
   执行以下命令可以查看当前的 **PATH** 设置：

```
$ echo $PATH
```

安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
$ oc <command>
```

13.1.7.2. 在 Windows 上安装 OpenShift CLI

您可以按照以下流程在 Windows 上安装 OpenShift CLI（**oc**）二进制代码。

流程

1. 进入到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 在 **Version** 下拉菜单中选择相应的版本。
3. 单击 **OpenShift v4.6 Windows 客户端**条目旁边的 **Download Now**，再保存文件。
4. 使用 ZIP 程序解压存档。
5. 把 **oc** 二进制代码放到 **PATH** 中的目录中。
   要查看您的 **PATH**，请打开命令提示窗口并执行以下命令：

```
C:\> path
```

安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
C:\> oc <command>
```

13.1.7.3. 在 macOS 上安装 OpenShift CLI

您可以按照以下流程在 macOS 上安装 OpenShift CLI（**oc**）二进制代码。

流程

1. 进入到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 在 **Version** 下拉菜单中选择相应的版本。
3. 单击 **OpenShift v4.6 MacOSX 客户端**条目旁边的 **Download Now**，再保存文件。
4. 解包和解压存档。

5. 将 **oc** 二进制文件移到 PATH 的目录中。
   要查看您的 **PATH**，打开一个终端窗口并执行以下命令：

```
$ echo $PATH
```

安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
$ oc <command>
```

13.1.8. 手动创建安装配置文件

对于使用用户置备的基础架构的 OpenShift Container Platform 安装，您必须手动生成安装配置文件。

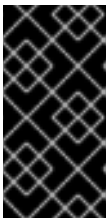
先决条件

- 获取 OpenShift Container Platform 安装程序和集群的访问令牌。

流程

1. 创建用来存储您所需的安装资产的安装目录：

```
$ mkdir <installation_directory>
```



重要

您必须创建目录。一些安装信息，如 bootstrap X.509 证书，有较短的过期间隔，因此不要重复使用安装目录。如果要重复使用另一个集群安装中的个别文件，可以将其复制到您的目录中。但是，一些安装数据的文件名可能会在发行版本之间有所改变。从 OpenShift Container Platform 老版本中复制安装文件时要格外小心。

2. 自定义以下 **install-config.yaml** 文件模板，并将它保存到 **<installation_directory>** 中。

注意

此配置文件必须命名为 **install-config.yaml**。

3. 备份 **install-config.yaml** 文件，以便用于安装多个集群。

重要

install-config.yaml 文件会在安装过程的下一步骤中消耗掉。现在必须备份它。

您可以自定义 **install-config.yaml** 文件，以指定有关 OpenShift Container Platform 集群平台的更多信息，或修改所需参数的值。

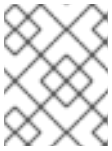
```
apiVersion: v1
baseDomain: example.com (1)
compute: (2)
- hyperthreading: Enabled (3)
  name: worker
```

```
  replicas: 0 4
controlPlane: 5
  hyperthreading: Enabled 6
  name: master
  replicas: 3 7
metadata:
  name: test 8
networking:
  clusterNetwork:
  - cidr: 10.128.0.0/14 9
    hostPrefix: 23 10
  networkType: OpenShiftSDN
  serviceNetwork: 11
  - 172.30.0.0/16
platform:
  none: {} 12
fips: false 13
pullSecret: '{"auths": ...}' 14
sshKey: 'ssh-ed25519 AAAA...' 15
```

1 集群的基域。所有 DNS 记录都必须是这个基域的子域，并包含集群名称。

2 5 **controlPlane** 部分是一个单映射，但 **compute** 部分是一系列映射。为满足不同数据结构的要求，**compute** 部分的第一行必须以连字符 **-** 开头，**controlPlane** 部分的第一行则不可以连字符开头。只使用一个 control plane 池。

3 6 是否要启用或禁用并发多线程（SMT）或**超线程**。默认情况下，启用 SMT 可提高机器内核的性能。您可以通过将参数值设为 **Disabled** 来禁用。如果禁用 SMT，则必须在所有集群机器中禁用它，其中包括 control plane 和计算机器。



注意

默认启用并发多线程（SMT）。如果在 BIOS 设置中没有启用 SMT，**hyperthreading** 参数不会起作用。

重要

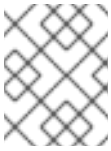
如果您禁用 **hyperthreading**（无论是在 BIOS 中还是在 **install-config.yaml** 中），请确保您对可能会造成的机器性能显著降低的情况有所考虑。

4 **replicas** 参数的值必须设置为 **0**。此参数控制集群为您创建和管理的 worker 数量，使用用户置备的基础架构时集群不会执行这些功能。在完成 OpenShift Container Platform 安装前，您必须手动为集群部署 worker 机器。

7 您添加到集群的 control plane 机器数量。由于集群将这个值用作集群中 etcd 端点的数量，因此该值必须与您部署的 control plane 机器数量匹配。

8 您在 DNS 记录中指定的集群名称。

9 从中分配 pod IP 地址的 IP 地址块。此块不得与现有的物理网络重叠。这些 IP 地址用于 pod 网络。如果您需要从外部网络访问 pod，请配置负载均衡器和路由器来管理流量。



注意

类 E CIDR 范围保留给以后使用。要使用 Class E CIDR 范围，您必须确保您的网络环境接受 Class E CIDR 范围内的 IP 地址。

10 分配给每个单独节点的子网前缀长度。例如，如果 **hostPrefix** 设为 **23**，则每个节点从所给的 **cidr** 中分配一个 **/23** 子网，这样就能有 510 (2^(32 - 23) - 2) 个 Pod IP 地址。如果您需要从外部网络访问节点，请配置负载均衡器和路由器来管理流量。

11 用于服务 IP 地址的 IP 地址池。您只能输入一个 IP 地址池。此块不得与现有的物理网络重叠。如果您需要从外部网络访问服务，请配置负载均衡器和路由器来管理流量。

12 您必须将平台设置为 **none**。您不能为您的平台提供额外的平台配置变量。

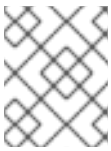
13 是否启用或禁用 FIPS 模式。默认情况下不启用 FIPS 模式。如果启用了 FIPS 模式，运行 OpenShift Container Platform 的 Red Hat Enterprise Linux CoreOS (RHCOS) 机器会绕过默认的 Kubernetes 加密套件，并使用由 RHCOS 提供的加密模块。

重要

只有在 **x86_64** 架构中的 OpenShift Container Platform 部署支持 FIPS 验证的/Modules in Process 加密库。

14 Red Hat OpenShift Cluster Manager 中的 pull secret 。通过此 pull secret，您可以进行所含授权机构提供的服务的身份验证，这些服务包括为 OpenShift Container Platform 组件提供容器镜像的 Quay.io。

15 Red Hat Enterprise Linux CoreOS (RHCOS) 中 **core** 用户的默认 SSH 密钥的公钥部分。



注意

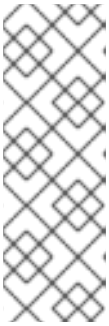
对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 **ssh-agent** 进程使用的 SSH 密钥。

13.1.8.1. 在安装过程中配置集群范围代理

生产环境可能会拒绝直接访问互联网，而是提供 HTTP 或 HTTPS 代理。您可以通过在 **install-config.yaml** 文件中配置代理设置，将新的 OpenShift Container Platform 集群配置为使用代理。

先决条件

- 您有一个现有的 **install-config.yaml** 文件。
- 您检查了集群需要访问的站点，并决定是否需要绕过代理。默认情况下代理所有集群出口流量，包括对托管云供应商 API 的调用。您需要将站点添加到 **Proxy** 对象的 **spec.noProxy** 字段来绕过代理。



注意

Proxy 对象 **status.noProxy** 字段使用安装配置中的 **networking.machineNetwork[].cidr**、**networking.clusterNetwork[].cidr** 和 **networking.serviceNetwork[]** 字段的值填充。

对于在 Amazon Web Services(AWS)、Google Cloud Platform(GCP)、Microsoft Azure 和 Red Hat OpenStack Platform(RHOSP)上安装, **Proxy** 对象 **status.noProxy** 字段也会使用实例元数据端点填充(**169.254.169.254**)。

流程

1. 编辑 **install-config.yaml** 文件并添加代理设置。例如：

```
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: example.com 3
additionalTrustBundle: | 4
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
...
```

1. 用于创建集群外 HTTP 连接的代理 URL。URL 必须是 **http**。
2. 用于创建集群外 HTTPS 连接的代理 URL。
3. 要排除在代理中的目标域名、IP 地址或其他网络 CIDR 的逗号分隔列表。在域前面加 **.** 来仅匹配子域。例如： **.y.com** 匹配 **x.y.com**，但不匹配 **y.com**。使用 * 绕过所有目的地的代理。
4. 如果提供，安装程序会在 **openshift-config 命名空间中生成名为 user-ca- bundle** 的配置映射来保存额外的 CA 证书。如果您提供 **additionalTrustBundle** 和至少一个代理设置，则 **Proxy** 对象会被配置为引用 **trustedCA** 字段中的 **user-ca-bundle** 配置映射。然后，Cluster Network Operator 会创建一个 **trusted-ca-bundle** 配置映射，该配置映射将为 **trustedCA** 参数指定的内容与 RHCOS 信任捆绑包合并。 **additionalTrustBundle** 字段是必需的，除非代理的身份证书由来自 RHCOS 信任捆绑包的颁发机构签名。

注意

安装程序不支持代理的 **readinessEndpoints** 字段。

2. 保存该文件，并在安装 OpenShift Container Platform 时引用。

安装程序会创建一个名为 **cluster** 的集群范围代理，该代理使用提供的 **install-config.yaml** 文件中的代理设置。如果没有提供代理设置，仍然会创建一个 **cluster Proxy** 对象，但它会有一个空 **spec**。

注意

只支持名为 **cluster** 的 **Proxy** 对象，且无法创建额外的代理。

13.1.9. 配置三节点集群

您可在没有 worker 的 OpenShift Container Platform 中安装和运行三节点集群。这为集群管理员和开发人员提供了较小的、效率更高的集群，用于开发、生产及测试。

流程

- 编辑 **install-config.yaml** 文件，将计算副本（也称为 worker 副本）数设为 **0**，如以下 **compute** 小节中所示：

```
compute:
- name: worker
  platform: {}
  replicas: 0
```

13.1.10. 创建 Kubernetes 清单和 Ignition 配置文件

由于您必须修改一些集群定义文件并要手动启动集群机器，因此您必须生成 Kubernetes 清单和 Ignition 配置文件，集群需要这两项来创建其机器。

安装配置文件转换为 Kubernetes 清单。清单嵌套到 Ignition 配置文件中，稍后用于创建集群。

重要

- 安装程序生成的 Ignition 配置文件包含在 24 小时后过期的证书，然后在过期时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外情况是，您需要手动批准待处理的 **node-bootstrapper** 证书签名请求（CSR）来恢复 kubelet 证书。如需更多信息，请参阅*从过期的 control plane 证书中恢复*的文档。
- 建议您在生成 12 小时后使用 Ignition 配置文件，因为集群安装后 24 小时证书从 16 小时轮转至 22 小时。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中运行证书更新时避免安装失败。

先决条件

- 已获得 OpenShift Container Platform 安装程序。
- 已创建 **install-config.yaml** 安装配置文件。

流程

1. 切换到包含安装程序的目录，并为集群生成 Kubernetes 清单：

```
$ ./openshift-install create manifests --dir <installation_directory> 1
```

1 对于 **<installation_directory>**，请指定含有您创建的 **install-config.yaml** 文件的安装目录。

警告

如果要安装一个三节点集群，请跳过以下步骤，以便 control plane 节点可以调度。

+

重要

当您将 control plane 节点从默认的不可调度配置为可以调度时，需要额外的订阅。这是因为 control plane 节点随后变为 worker 节点。

1. 检查 **<installation_directory>/manifests/cluster-scheduler-02-config.yml** Kubernetes 清单文件中的 **mastersSchedulable** 参数是否已设置为 **false**。此设置可防止在 control plane 机器上调度 pod:
   a. 打开 **<installation_directory>/manifests/cluster-scheduler-02-config.yml** 文件。
   b. 找到 **mastersSchedulable** 参数并确保它被设置为 **false**。
   c. 保存并退出文件。
2. 要创建 Ignition 配置文件，从包含安装程序的目录运行以下命令：

```
$ ./openshift-install create ignition-configs --dir <installation_directory> 1
```

1 对于 **<installation_directory>**，请指定相同的安装目录。

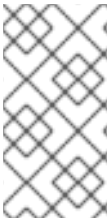
该目录中将生成以下文件：

```
.
├── auth
│   ├── kubeadmin-password
│   └── kubeconfig
├── bootstrap.ign
├── master.ign
├── metadata.json
└── worker.ign
```

13.1.11. 安装 RHCOS 并启动 OpenShift Container Platform bootstrap 过程

要在您置备的裸机基础架构上安装 OpenShift Container Platform，您必须在机器上安装 Red Hat Enterprise Linux CoreOS（RHCOS）。安装 RHCOS 时，您必须为 OpenShift Container Platform 安装程序生成的机器类型提供 Ignition 配置文件。如果您配置了合适的网络、DNS 和负载均衡基础架构，OpenShift Container Platform bootstrap 过程会在 RHCOS 机器重启后自动开始。

要在机器上安装 RHCOS，请按照以下步骤使用 ISO 镜像或网络 PXE 启动。



注意

本安装文档中包括的计算节点部署步骤特定于 RHCOS。如果您选择部署基于 RHEL 的计算节点，您将接管所有操作系统生命周期管理和维护，包括执行系统更新、应用补丁和完成所有其他必要的任务。RHEL 7 计算机器的使用已弃用，计划在以后的 OpenShift Container Platform 4 发行版本中删除。

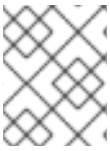
您可以使用以下方法在 ISO 和 PXE 安装过程中配置 RHCOS:

- 内核参数： 您可以使用内核参数来提供特定于安装的信息。例如，您可以指定上传到 HTTP 服务器的 RHCOS 安装文件的位置，以及您要安装的节点类型的 Ignition 配置文件的位置。对于 PXE 安装，您可以使用 **APPEND** 参数将参数传递给实时安装程序的内核。对于 ISO 安装，您可以中断实时安装引导过程来添加内核参数。在这两种安装情况下，您可以使用特殊的 **coreos.inst.*** 参数来指示实时安装程序，以及标准安装引导参数来打开或关闭标准内核服务。

- Ignition 配置：OpenShift Container Platform Ignition 配置文件（***.ign**）特定于您要安装的节点类型。您可以在 RHCOS 安装过程中传递 bootstrap、control plane 或计算节点 Ignition 配置文件的位置，以便在第一次引导时生效。特殊情况下，您可以创建单独的、有限的 Ignition 配置来传递给 Live 系统。该 Ignition 配置可以执行特定任务，如在安装完成后向置备系统报告成功。这个特殊 Ignition 配置由 **coreos-installer** 使用，用于首次启动安装的系统。不要直接向 live ISO 提供标准 control plane 和计算节点 Ignition 配置。

- **coreos-installer**：您可以将 live ISO 安装程序引导到 shell 提示符，这可让您在首次引导前以多种方式准备持久性系统。特别是，您可以运行 **coreos-installer** 命令来识别包括的工件、使用磁盘分区以及设置联网。在有些情况下，您可以配置 live 系统上的功能并将其复制到安装的系统中。

使用 ISO 安装还是 PXE 安装要根据您的具体情况而定。PXE 安装需要可用的 DHCP 服务并进行更多准备，但可以使安装过程更自动化。ISO 安装是一个更手动的过程，如果您设置的机器较多，则可能不方便。



注意

自 OpenShift Container Platform 4.6 起，RHCOS ISO 和其他安装工件支持在带有 4K 扇区的磁盘上安装。

13.1.11.1. 使用 ISO 镜像创建 Red Hat Enterprise Linux CoreOS (RHCOS) 机器

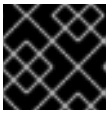
在您置备的基础架构上安装集群前，必须先创建 RHCOS 机器供其使用。您可以使用 ISO 镜像来创建这些机器。

先决条件

- 获取集群的 Ignition 配置文件。
- 具有可从计算机以及您创建的机器访问的 HTTP 服务器的访问权限。

流程

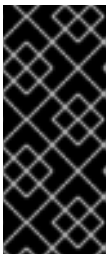
1. 将安装程序创建的 control plane、计算和 bootstrap Ignition 配置文件上传到 HTTP 服务器。记下这些文件的 URL。



重要

如果您计划在安装完成后在集群中添加更多计算机器，请不要删除这些文件。

2. 从 RHCOS 镜像镜像 页面获取您选择的操作系统实例安装方法所需的 RHCOS 镜像。



重要

RHCOS 镜像可能不会随着 OpenShift Container Platform 的每一发行版本都有改变。您必须下载最高版本的镜像，其版本号应小于或等于您安装的 OpenShift Container Platform 版本。如果可用，请使用与 OpenShift Container Platform 版本匹配的镜像版本。此流程只使用 ISO 镜像。此安装类型不支持 RHCOS qcow2 镜像。

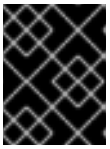
ISO 文件名类似以下示例：

rhcos-<version>-live.<architecture>.iso

3. 使用 ISO 启动 RHCOS 安装。使用如下安装选项之一：
   - 将 ISO 镜像刻录到磁盘并直接启动。
   - 通过 LOM 接口使用 ISO 重定向。
4. 引导 ISO 镜像。您可以中断安装引导过程来添加内核参数。然而，在这个 ISO 过程中，您应该使用 **coreos-installer** 命令而不是添加内核参数。如果您在没有选项或中断的情况下运行 live 安装程序，安装程序将引导至 live 系统上的 shell 提示符，准备好将 RHCOS 安装到磁盘中。
5. 在运行 **coreos-installer** 前，请参阅 *高级 RHCOS 安装参考* 部分，以了解配置功能的不同方法，如网络和磁盘分区。
6. 运行 **coreos-installer** 命令。您至少必须识别节点类型的 Ignition 配置文件位置，以及您要安装到的磁盘位置。下面是一个示例：

```
$ sudo coreos-installer install \
    --ignition-url=https://host/worker.ign /dev/sda
```

7. 安装 RHCOS 后，系统会重启。系统重启过程中，它会应用您指定的 Ignition 配置文件。
8. 继续为集群创建其他机器。



重要

此刻您必须创建 bootstrap 和 control plane 机器。如果 control plane 机器不可调度（这是默认调度），则在安装集群前至少会创建两台计算机器。

13.1.11.2. 通过 PXE 或 iPXE 启动来创建 Red Hat Enterprise Linux CoreOS (RHCOS) 机器

在安装使用手动置备 RHCOS 节点（如裸机）的集群前，您必须创建 RHCOS 机器供其使用。您可以使用 PXE 或 iPXE 启动来创建机器。

先决条件

- 获取集群的 Ignition 配置文件。
- 配置合适的 PXE 或 iPXE 基础架构。
- 具有 HTTP 服务器的访问权限，以便您可从计算机进行访问。

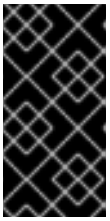
流程

1. 将安装程序创建的 master、worker 和 bootstrap Ignition 配置文件上传到 HTTP 服务器。记下这些文件的 URL。

重要

您可以在 Ignition 配置中添加或更改配置设置，然后将其保存到 HTTP 服务器。如果您计划在安装完成后在集群中添加更多计算机器，请不要删除这些文件。

2. 从 RHCOS 镜像 镜像页面获取 RHCOS **内核、initram fs 和 rootfs** 文件。



重要

RHCOS 工件（artifact）可能不会随着 OpenShift Container Platform 的每个发行版本而改变。您必须下载最高版本的工件，其版本号应小于或等于您安装的 OpenShift Container Platform 版本。这个过程只使用下面描述的正确 **kernel**、**initramfs** 和 **rootfs** 工件。此安装类型不支持 RHCOS qcow2 镜像。

文件名包含 OpenShift Container Platform 版本号。它们类似以下示例：

- **kernel**: **rhcos-<version>-live-kernel-<architecture>**
- **initramfs**: **rhcos-<version>-live-initramfs.<architecture>.img**
- **rootfs**: **rhcos-<version>-live-rootfs.<architecture>.img**

3. 上传引导方法所需的额外文件：

- 对于传统的 PXE，将 **kernel** 和 **initramfs** 文件上传到 TFTP 服务器，并将 **rootfs** 文件上传到 HTTP 服务器。
- 对于 iPXE，将 **kernel**、**initram** fs 和 **rootfs** 文件上传到 HTTP 服务器。

重要

如果您计划在安装完成后在集群中添加更多计算机器，请不要删除这些文件。

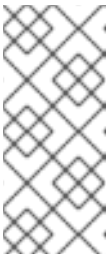
4. 配置网络启动基础架构，以便在安装 RHCOS 后机器可从本地磁盘启动。

5. 为 RHCOS 镜像配置 PXE 或 iPXE 安装。
针对您的环境修改以下示例菜单条目之一，并验证能否正确访问镜像和 Ignition 文件：

- 对于 PXE：

```
DEFAULT pxeboot
TIMEOUT 20
PROMPT 0
LABEL pxeboot
    KERNEL http://<HTTP_server>/rhcos-<version>-live-kernel-<architecture> (1)
    APPEND initrd=http://<HTTP_server>/rhcos-<version>-live-initramfs.
<architecture>.img coreos.live.rootfs_url=http://<HTTP_server>/rhcos-<version>-live-
rootfs.<architecture>.img coreos.inst.install_dev=/dev/sda
coreos.inst.ignition_url=http://<HTTP_server>/bootstrap.ign (2) (3)
```

1. 指定上传到 HTTP 服务器的 live **kernel** 文件位置。URL 必须是 HTTP、TFTP 或者 FTP ; 不支持 HTTPS 和 NFS。
2. 如果您使用多个 NIC，请在 **ip** 选项中指定一个接口。例如，要在名为 **eno1** 的 NIC 上使用 DHCP，请设置 **ip=eno1:dhcp**。
3. 指定上传到 HTTP 服务器的 RHCOS 文件的位置。**initrd** 参数值是 **initramfs** 文件的位置，**coreos.live.rootfs_url** 参数值是 **rootfs** 文件的位置，**coreos.inst.ignition_url** 参数值是 bootstrap Ignition 配置文件的位置。您还可以在 **APPEND** 行中添加更多内核参数来配置联网或其他引导选项。



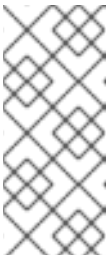
注意

这个配置不会在使用图形控制台的机器上启用串口控制台访问。要配置不同的控制台，请在 **APPEND** 行中添加一个或多个 **console=** 参数。例如，添加 **console=tty0 console=ttyS0** 将第一个 PC 串口设置为主控制台，图形控制台作为二级控制台。如需更多信息，请参阅如何在 Red Hat Enterprise Linux 中设置串行终端和（或）控制台？

- 对于 iPXE：

```
kernel http://<HTTP_server>/rhcos-<version>-live-kernel-<architecture> initrd=main
coreos.live.rootfs_url=http://<HTTP_server>/rhcos-<version>-live-rootfs.
<architecture>.img coreos.inst.install_dev=/dev/sda
coreos.inst.ignition_url=http://<HTTP_server>/bootstrap.ign 1 2
initrd --name main http://<HTTP_server>/rhcos-<version>-live-initramfs.
<architecture>.img 3
boot
```

1. 指定上传到 HTTP 服务器的 RHCOS 文件的位置。**kernel** 参数值是 **kernel** 文件的位置，在 UEFI 系统中引导时需要 **initrd=main** 参数。 **coreos.live.rootfs_url** 参数值是 **rootfs** 文件的位置， **coreos.inst.ignition_url** 参数值则是 bootstrap Ignition 配置文件的位置。
2. 如果您使用多个 NIC，请在 **ip** 选项中指定一个接口。例如，要在名为 **eno1** 的 NIC 上使用 DHCP，请设置 **ip=eno1:dhcp**。
3. 指定上传到 HTTP 服务器的 **initramfs** 文件的位置。



注意

这个配置不会在使用图形控制台的机器上启用串口控制台访问。要配置不同的控制台，请在 **kerne** 行中添加一个或多个 **console=** 参数。例如，添加 **console=tty0 console=ttyS0** 将第一个 PC 串口设置为主控制台，图形控制台作为二级控制台。如需更多信息，请参阅如何在 Red Hat Enterprise Linux 中设置串行终端和（或）控制台？

6. 如果使用 PXE UEFI，请执行以下操作：
   a. 提供引导系统所需的 **shim x64.efi 和 grubx64 .efi EFI 二进制文件以及 grub.cfg** 文件。
      - 通过将 RHCOS ISO 挂载到主机，然后将 **images/efiboot.img** 文件挂载到您的主机来提取所需的 EFI 二进制文件：

```
$ mkdir -p /mnt/iso
```

```
$ mkdir -p /mnt/efiboot
```

```
$ mount -o loop rhcos-installer.x86_64.iso /mnt/iso
```

```
$ mount -o loop,ro /mnt/iso/images/efiboot.img /mnt/efiboot
```

- 从 **efiboot.img** 挂载点，将 **EFI/redhat/shimx64.efi** 和 **EFI/redhat/grubx64.efi** 文件复制到 TFTP 服务器中：

```
$ cp /mnt/efiboot/EFI/redhat/shimx64.efi .
```

```
$ cp /mnt/efiboot/EFI/redhat/grubx64.efi .
```

```
$ umount /mnt/efiboot
```

```
$ umount /mnt/iso
```

- 将 RHCOS ISO 中包含的 **EFI/redhat/grub.cfg** 文件复制到您的 TFTP 服务器中。

b. 编辑 **grub.cfg** 文件使其包含类似如下的参数：

```
menuentry 'Install Red Hat Enterprise Linux CoreOS' --class fedora --class gnu-linux --
class gnu --class os {
 linuxefi rhcos-<version>-live-kernel-<architecture> coreos.inst.install_dev=/dev/sda
coreos.live.rootfs_url=http://<HTTP_server>/rhcos-<version>-live-rootfs.
<architecture>.img coreos.inst.ignition_url=http://<HTTP_server>/bootstrap.ign
 initrdefi rhcos-<version>-live-initramfs.<architecture>.img
}
```

其中：

rhcos-<version>-live-kernel-<architecture>

指定上传到 TFTP 服务器的 **内核** 文件。

http://<HTTP_server>/rhcos-<version>-live-rootfs.<architecture>.img

指定上传到 HTTP 服务器的 live rootfs 镜像的位置。

http://<HTTP_server>/bootstrap.ign

指定上传到 HTTP 服务器的 bootstrap Ignition 配置文件的位置。

rhcos-<version>-live-initramfs.<architecture>.img

指定上传到 TFTP 服务器的 **initramfs** 文件的位置。

注意

有关如何为 UEFI 引导配置 PXE 服务器的更多信息，请参阅红帽知识库文章：如何为 Red Hat Enterprise Linux 的 UEFI 引导配置/设置 PXE 服务器？

7. 继续为集群创建机器。

重要

此刻您必须创建 bootstrap 和 control plane 机器。如果 control plane 机器不可调度（这是默认调度），则在安装集群前至少会创建两台计算机器。

13.1.11.3. 高级 Red Hat Enterprise Linux CoreOS（RHCOS）安装配置

为 OpenShift Container Platform 手动置备 Red Hat Enterprise Linux CoreOS（RHCOS）节点的一个关键优点是能够进行通过默认的 OpenShift Container Platform 安装方法无法进行的配置。本节介绍了您可以使用的一些技术来进行配置，其中包括：

- 将内核参数传递给实时安装程序
- 从 live 系统手动运行 **coreos-installer**
- 将 Ignition 配置嵌入 ISO 中

本节详述了与 Red Hat Enterprise Linux CoreOS（RHCOS）手动安装的高级配置相关的内容，如磁盘分区、网络以及使用 Ignition 配置的不同方式相关。

13.1.11.3.1. 使用高级网络选项进行 PXE 和 ISO 安装

OpenShift Container Platform 节点的网络默认使用 DHCP 来收集所有必要配置设置。要设置静态 IP 地址或配置特殊的设置，如绑定,您可以执行以下操作之一：

- 引导 live 安装程序时会传递特殊的内核参数。
- 使用机器配置将网络文件复制到安装的系统中。
- 使用 live installer shell 提示配置网络，然后将那些设置复制到安装的系统上，以便在安装的系统第一次引导时生效。

要配置 PXE 或 iPXE 安装，请使用以下选项之一：

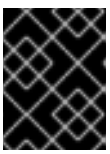
- 请参阅"高级 RHCOS 安装参考"表。
- 使用机器配置将网络文件复制到安装的系统中。

要配置 ISO 安装，请使用以下步骤。

流程

1. 引导 ISO 安装程序。
2. 在 live 系统 shell 提示下，使用可用的 RHEL 工具（如 **nmcli** 或 **nmtui**）为 Live 系统配置网络。
3. 运行 **coreos-installer** 命令来安装系统，添加 **--copy-network** 选项来复制网络配置。例如：

```
$ coreos-installer install --copy-network \
    --ignition-url=http://host/worker.ign /dev/sda
```



重要

copy-network 选项只复制 **/etc/NetworkManager/system-connections** 下的网络配置。特别是，它不会复制系统主机名。

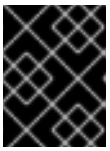
4. 重启安装的系统。

13.1.11.3.2. 磁盘分区

磁盘分区是在 Red Hat Enterprise Linux CoreOS（RHCOS）安装过程中在 OpenShift Container Platform 集群节点上创建的。特定架构的每个 RHCOS 节点都使用相同的分区布局，除非默认分区配置被覆盖。在 RHCOS 安装过程中，根文件系统的大小会增大，以使用目标设备中剩余的可用空间。

但是，在安装 OpenShift Container Platform 节点时，在两种情况下您可能需要覆盖默认分区：

- 创建单独的分区：对于在空磁盘中的 greenfield 安装，您可能想要在分区中添加单独的存储。这只在生成 **/var** 或者一个 **/var** 独立分区的子目录（如 **/var/lib/etcd** ）时被正式支持，但不支持两者。



重要

Kubernetes 只支持两个文件系统分区。如果您在原始配置中添加多个分区，Kubernetes 无法监控所有这些分区。

- 保留现有分区：对于 brownfield 安装，您要在现有节点上重新安装 OpenShift Container Platform，并希望保留从之前的操作系统中安装的数据分区，对于 **coreos-installer** 来说，引导选项和选项都允许您保留现有数据分区。

13.1.11.3.2.1. 创建一个独立的 /var 分区

通常情况下，OpenShift Container Platform 的磁盘分区应该留给安装程序。然而，在有些情况下您可能需要在文件系统的一部分中创建独立分区。

OpenShift Container Platform 支持添加单个分区来将存储附加到 **/var** 分区或 **/var** 的子目录。例如：

- **/var/lib/containers**：保存镜像相关的内容，随着更多镜像和容器添加到系统中，它所占用的存储会增加。
- **/var/lib/etcd**：保存您可能希望保持独立的数据，比如 etcd 存储的性能优化。
- **/var**：保存您希望独立保留的数据，用于特定目的（如审计）。

单独存储 **/var** 目录的内容可方便地根据需要对区域扩展存储，并可以在以后重新安装 OpenShift Container Platform 时保持该数据地完整。使用这个方法，您不必再次拉取所有容器，在更新系统时也无法复制大量日志文件。

因为 **/var** 在进行一个全新的 Red Hat Enterprise Linux CoreOS（RHCOS）安装前必需存在，所以这个流程会在 OpenShift Container Platform 安装过程的 **openshift-install** 准备阶段插入的机器配置来设置独立的 **/var** 分区。

流程

1. 创建存放 OpenShift Container Platform 安装文件的目录：

```
$ mkdir $HOME/clusterconfig
```

2. 运行 **openshift-install** 在 **manifest** 和 **openshift** 子目录中创建一组文件。在出现提示时回答系统问题：

```
$ openshift-install create manifests --dir $HOME/clusterconfig
```

```
? SSH Public Key ...
$ ls $HOME/clusterconfig/openshift/
99_kubeadmin-password-secret.yaml
99_openshift-cluster-api_master-machines-0.yaml
99_openshift-cluster-api_master-machines-1.yaml
99_openshift-cluster-api_master-machines-2.yaml
...
```

3. 创建 **MachineConfig** 对象并将其添加到 **openshift** 目录中的一个文件中。例如，把文件命名为 **98-var-partition.yaml**，将磁盘设备名称改为 **worker** 系统中存储设备的名称，并根据情况设置存储大小。这个示例将 **/var** 目录放在一个单独的分区中：

```
apiVersion: machineconfiguration.openshift.io/v1
kind: MachineConfig
metadata:
  labels:
    machineconfiguration.openshift.io/role: worker
  name: 98-var-partition
spec:
  config:
    ignition:
      version: 3.1.0
    storage:
      disks:
      - device: /dev/<device_name> 1
        partitions:
        - label: var
          startMiB: <partition_start_offset> 2
          sizeMiB: <partition_size> 3
      filesystems:
        - device: /dev/disk/by-partlabel/var
          path: /var
          format: xfs
    systemd:
      units:
        - name: var.mount 4
          enabled: true
          contents: |
            [Unit]
            Before=local-fs.target
            [Mount]
            What=/dev/disk/by-partlabel/var
            Where=/var
            Options=defaults,prjquota 5
            [Install]
            WantedBy=local-fs.target
```

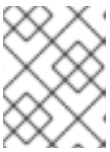
1. 要分区的磁盘的存储设备名称。

2. 当在引导磁盘中添加数据分区时，推荐最少使用 25000MB。root 文件系统会自动重新定义大小使其占据所有可用空间（最多到指定的偏移值）。如果没有指定值，或者指定的值小于推荐的最小值，则生成的 root 文件系统会太小，而在以后进行的 RHCOS 重新安装可能会覆盖数据分区的开始部分。

3. 数据分区的大小（以兆字节为单位）。

4 挂载单元的名称必须与 **Where=** 指令中指定的目录匹配。例如，对于挂载在 **/var/lib/containers** 上的文件系统，该单元必须命名为 **var-lib-containers.mount**。

5 对于用于容器存储的文件系统，必须启用 **prjquota** 挂载选项。



注意

在创建独立 **/var** 分区时，如果不同的实例类型没有相同的设备名称，则无法将不同的实例类型用于 worker 节点。

4. 再次运行 **openshift-install**，从 **manifest** 和 **openshift** 子目录中的一组文件创建 Ignition 配置：

```
$ openshift-install create ignition-configs --dir $HOME/clusterconfig
$ ls $HOME/clusterconfig/
auth  bootstrap.ign  master.ign  metadata.json  worker.ign
```

现在，可以使用 Ignition 配置文件作为 ISO 或 PXE 手动安装过程的输入来安装 Red Hat Enterprise Linux CoreOS（RHCOS）系统。

13.1.11.3.2.2. 保留现有分区

对于 ISO 安装，您可以在 **coreos-installer** 命令行中添加可让安装程序维护一个或多个现有分区的选项。对于 PXE 安装，您可以 **APPEND coreos.inst.*** 选项来保留分区。

保存的分区可能是来自现有 OpenShift Container Platform 系统中的分区，其中包括了您希望保留的数据分区。以下是几个提示：

- 如果您保存了现有分区，且这些分区没有为 RHCOS 留下足够空间，则安装将失败但不会损害已保存的分区。
- 通过分区标签或数字识别您要保留的磁盘分区。

对于 ISO 安装

这个示例保留分区标签以**数据（data*）**开头的任何分区：

```
# coreos-installer install --ignition-url http://10.0.2.2:8080/user.ign \
      --save-partlabel 'data*' /dev/sda
```

以下示例演示了在运行 **coreos-installer** 时要保留磁盘上的第 6 个分区：

```
# coreos-installer install --ignition-url http://10.0.2.2:8080/user.ign \
      --save-partindex 6 /dev/sda
```

这个示例保留了分区 5 及更高分区：

```
# coreos-installer install --ignition-url http://10.0.2.2:8080/user.ign
      --save-partindex 5- /dev/sda
```

在前面已保存分区的示例中，**coreos-installer** 会立即重新创建分区。

对于 PXE 安装

这个 **APPEND** 选项保留分区标签以 'data'('data*')开头的所有分区：

```
coreos.inst.save_partlabel=data*
```

这个 **APPEND** 选项保留分区 5 及其后的分区：

```
coreos.inst.save_partindex=5-
```

这个 **APPEND** 选项保留分区 6:

```
coreos.inst.save_partindex=6
```

13.1.11.3.3. 标识 Ignition 配置

在进行 RHCOS 手动安装时，您可以提供两种 Ignition 配置类型，它们有不同的原因：

- **永久安装 Ignition 配置**：每个手动 RHCOS 安装都需要传递 **openshift-installer** 生成的 Ignition 配置文件之一，如 **bootstrap.ign**、**master.ign** 和 **worker.ign**，才能进行安装。

  重要

  不建议修改这些文件。

  对于 PXE 安装，您可以使用 **coreos.inst.ignition_url=** 选项在 **APPEND** 行上传递 Ignition 配置。对于 ISO 安装，在 ISO 引导至 shell 提示符后，您可以使用 **--ignition-url=** 选项在 **coreos-installer** 命令行上识别 Ignition 配置。在这两种情况下，都只支持 HTTP 和 HTTPS 协议。

- **live 安装 Ignition 配置**：此类型必须手动创建，并应该尽可能避免，因为红帽不支持它。使用此方法，Ignition 配置会传递到 live 安装介质，在引导时立即运行，并在 RHCOS 系统安装到磁盘之前和/或之后执行设置任务。这个方法只用于必须执行一次且之后不能再次应用的任务，如不能使用机器配置进行的高级分区。
  对于 PXE 或 ISO 引导，您可以创建 Ignition 配置，**APPEND ignition.config.url=** 选项，以标识 Ignition 配置的位置。您还需要附加 **ignition.firstboot ignition.platform.id=metal** 或者 **ignition.config.url** 选项。

13.1.11.3.3.1. 在 RHCOS ISO 中嵌入 Ignition 配置

您可以直接嵌入 RHCOS ISO 镜像中的 live 安装 Ignition 配置。引导 ISO 镜像后，内嵌的配置将自动应用。

流程

1. 从以下镜像镜像页面下载 **coreos-installer** 二进制文件：
   https://mirror.openshift.com/pub/openshift-v4/clients/coreos-installer/latest/。
2. 检索 RHCOS ISO 镜像和 Ignition 配置文件，并将其复制到可访问的目录中，如 **/mnt**:

   ```
   # cp rhcos-<version>-live.x86_64.iso bootstrap.ign /mnt/
   # chmod 644 /mnt/rhcos-<version>-live.x86_64.iso
   ```

3. 运行以下命令将 Ignition 配置嵌入 ISO 中：

   ```
   # ./coreos-installer iso ignition embed -i /mnt/bootstrap.ign \
       /mnt/rhcos-<version>-live.x86_64.iso
   ```

现在，您以使用该 ISO 使用指定的 live 安装 Ignition 配置来安装 RHCOS。

重要

不支持且不推荐使用 **coreos-installer iso ignition embed** 来嵌入由 **openshift-installer** 生成的文件，如 **bootstrap.ign**、**master.ign** 和 **worker.ign**。

4. 要显示嵌入的 Ignition 配置的内容并将其定向到文件中，请运行：

```
# ./coreos-installer iso ignition show /mnt/rhcos-<version>-live.x86_64.iso > mybootstrap.ign
```

```
# diff -s bootstrap.ign mybootstrap.ign
```

输出示例

```
Files bootstrap.ign and mybootstrap.ign are identical
```

5. 要删除 Ignition 配置并将 ISO 返回到其 pristine 状态（因此您可以重复使用它），请运行：

```
# ./coreos-installer iso ignition remove /mnt/rhcos-<version>-live.x86_64.iso
```

现在，您可以将另一个 Ignition 配置嵌入到 ISO 中，或者在其 pristine 状态下使用 ISO。

13.1.11.3.4. 高级 RHCOS 安装参考

本节演示了网络配置和其他高级选项，允许您修改 Red Hat Enterprise Linux CoreOS（RHCOS）手动安装过程。下表描述了您可以与 RHCOS live installer 和 **coreos-installer** 命令一起使用的内核参数和命令行选项。

RHCOS 启动提示下的路由和绑定选项
如果从 ISO 镜像安装 RHCOS，您可以在引导该镜像时手动添加内核参数以配置节点的网络。如果没有使用网络参数，则安装默认为使用 DHCP。

重要

添加网络参数时，还必须添加 **rd.neednet=1** 内核参数。

下表描述了如何为实时 ISO 安装使用 **ip=** 、 **nameserver=** 和 **bond=** 内核参数。

注意

在添加内核参数时顺序非常重要： **ip=**，**nameserver=**，然后 **bond=**。

ISO 的路由和绑定选项

下表提供了配置 Red Hat Enterprise Linux CoreOS（RHCOS）节点网络的示例。这些是在系统引导过程中传递给 **dracut** 工具的网络选项。有关 **dracut** 支持的网络选项的详情，请参考 **dracut.cmdline** 手册页。

描述	例子
要配置一个 IP 地址，可以使用 DHCP(**ip=dhcp**)或者设置单独的静态 IP 地址(**ip=<host_ip>**)。然后在每个节点上指定 DNS 服务器 IP 地址(**nameserver=<dns_ip>**)。这个示例设置： • 节点的 IP 地址为 **10.10.10.2** • 网关地址为 **10.10.10.254** • 子网掩码为 **255.255.255.0** • 主机名为 **core0.example.com** • DNS 服务器地址为 **4.4.4.41**	ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp1s0:none nameserver=4.4.4.41
通过指定多个 **ip=** 条目来指定多个网络接口。	ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp1s0:none ip=10.10.10.3::10.10.10.254:255.255.255.0:core0.example.com:enp2s0:none
可选：您可以通过设置一个 **rd.route=** 值来配置到额外网络的路由。 如果额外网络网关与主要网络网关不同，则默认网关必须是主要网络网关。	配置默认网关： ip=::10.10.10.254:::: 为额外网络配置路由： rd.route=20.20.20.0/24:20.20.20.254:enp2s0
在单一接口中禁用 DHCP，比如当有两个或者多个网络接口时，且只有一个接口被使用。	ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp1s0:none ip=::::core0.example.com:enp2s0:none
您可以将系统中 DHCP 和静态 IP 配置与多个网络接口结合在一起。	ip=enp1s0:dhcp ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp2s0:none

描述	例子
可选： 您可以使用 **vlan=** 参数在单独的接口上配置 VLAN。	在网络接口中配置 VLAN 并使用静态 IP 地址： ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp2s0.100:none vlan=enp2s0.100:enp2s0 在网络接口中配置 VLAN 并使用 DHCP： ip=enp2s0.100:dhcp vlan=enp2s0.100:enp2s0
您可以为每个服务器添加一个 **nameserver=** 条目来提供多个 DNS 服务器。	nameserver=1.1.1.1 nameserver=8.8.8.8
可选：使用 **bond=** 选项支持将多个网络接口绑定到一个接口。在这两个示例中： - 配置绑定接口的语法为：**bond=name[:network_interfaces][:options]** - *name* 是绑定设备名称（**bond0**），*network_interfaces* 代表用逗号分开的物理（以太网）接口(**em1,em2**)的列表，*options* 是用逗号分开的绑定选项列表。输入 **modinfo bonding** 查看可用选项。 - 当使用 **bond=** 创建绑定接口时，您必须指定如何分配 IP 地址以及绑定接口的其他信息。	要将绑定的接口配置为使用 DHCP，请将绑定的 IP 地址设置为 **dhcp**。例如： bond=bond0:em1,em2:mode=active-backup ip=bond0:dhcp 要将绑定接口配置为使用静态 IP 地址，请输入您需要的特定 IP 地址以及相关信息。例如： bond=bond0:em1,em2:mode=active-backup ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:bond0:none
可选： 您可以使用 **vlan=** 参数在绑定接口上配置 VLAN。	使用 VLAN 配置绑定接口并使用 DHCP： ip=bond0.100:dhcp bond=bond0:em1,em2:mode=active-backup vlan=bond0.100:bond0 使用 VLAN 配置绑定接口，并使用静态 IP 地址： ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:bond0.100:none bond=bond0:em1,em2:mode=active-backup vlan=bond0.100:bond0

描述	例子
可选：通过使用 **team=** 参数，网络合作可用作绑定的替代选择。在此例中： • 配置组接口的语法为：**team=name[:network_interfaces]** *name* 是组设备名称(**team0**)，**network** *_interfaces* 代表以逗号分隔的物理（以太网）接口（**em1**、**em2**）列表。  **注意** 当 RHCOS 切换到即将推出的 RHEL 版本时，团队计划会被弃用。如需更多信息，请参阅红帽知识库文章。	配置网络团队： team=team0:em1,em2 ip=team0:dhcp

coreos.inst 引导选项用于 ISO 或 PXE 安装

虽然您可以将大多数标准安装引导参数传递给 live 安装程序，但也有一些特定于 RHCOS live 安装程序的参数。

- 对于 ISO，可以通过中断 RHCOS 安装程序来添加这些选项。
- 对于 PXE 或 iPXE，这些选项必须在启动 PXE 内核前添加到 **APPEND** 行中。您无法中断实时 PXE 安装。

下表显示了用于 ISO 和 PXE 安装的 RHCOS live installer 引导选项。

表 13.8. coreos.inst 引导选项

参数	描述
coreos.inst.install_dev	必需。要安装的系统中的块设备。虽然可以使用 **sda** 这样的相对路径，但建议使用完整路径，如 **/dev/sda**。
coreos.inst.ignition_url	可选：嵌入到已安装系统中的 Ignition 配置的 URL如果没有指定 URL，则不会嵌入 Ignition 配置。
coreos.inst.save_partlabel	可选：在安装过程中要保留的分区压缩标签。允许使用 glob 风格的通配符。指定分区不需要存在。
coreos.inst.save_partindex	可选：在安装过程中完成要保留的分区的分离索引。可以使用 **m-n** 指定范围，**m** 或 **n** 可以被省略。指定分区不需要存在。
coreos.inst.insecure	可选：将 **coreos.inst.image_url** 指定的 OS 镜像提交取消签名。

参数	描述
coreos.inst.image_url	可选：下载并安装指定的 RHCOS 镜像。 ● 这个参数不应该在生产环境中使用，而是只用于调试目的。 ● 虽然在 RHCOS 的安装版本与 live 介质的版本不匹配时可以使用这个参数，但建议使用与您要安装版本匹配的介质。 ● 如果您使用的是 **coreos.inst.image_url**。还必须使用 **coreos.inst.insecure**。这是因为，裸机介质没有为 OpenShift Container Platform 进行 GPG 签名。 ● 只支持 HTTP 和 HTTPS 协议。
coreos.inst.skip_reboot	可选：安装后该系统不会重启。安装完成后，您会收到提示，提示您检查在安装过程中发生的情况。这个参数不应该在生产环境中使用，而是只用于调试目的。
coreos.inst.platform_id	可选：安装 RHCOS 镜像的平台的 Ignition 平台 ID。默认为 **metal**。这个选项决定是否从云供应商（如 VMware）请求 Ignition 配置。例如：**coreos.inst.platform_id=vmware**。
ignition.config.url	可选：用于实时启动的 Ignition 配置的 URL。例如，它可以用来定制调用 **coreos-installer** 的方式，或者用来在安装前或安装后运行代码。这与 **coreos.inst.ignition_url**（这是已安装系统的 Ignition 配置）不同。

ISO 安装的 coreos-installer 选项

您还可以直接从命令行调用 **coreos-installer** 命令来安装 RHCOS。上表中的内核参数提供了在引导时自动调用 **coreos-installer** 的快捷方式，但您可以在 shell 提示符运行时将类似的参数直接传递给 **coreos-installer**。

下表显示了您可以在实时安装过程中从 shell 提示符传递给 **coreos-installer** 命令的选项和子命令。

表 13.9. CoreOS-installer 命令行选项、参数和子命令

命令行选项	
选项	**描述**
-u, --image-url <url>	手动指定镜像 URL。
-f, --image-file <path>	手动指定本地镜像文件。
-i, --ignition-file <path>	从文件中嵌入 Ignition 配置。

-I, --ignition-url <URL>	从 URL 嵌入 Ignition 配置。
--ignition-hash <digest>	Ignition config 的 **type-value** 的文摘值。
-p, --platform <name>	覆盖 Ignition 平台 ID。
--append-karg <arg>…	附加默认内核参数。
--delete-karg <arg>…	删除默认内核参数。
-n, --copy-network	从安装环境中复制网络配置。  **重要** **copy-network** 选项只复制 **/etc/NetworkManager/system-connections** 下的网络配置。特别是，它不会复制系统主机名。
--network-dir <path>	使用 **-n**。默认为 **/etc/NetworkManager/system-connections/**。
--save-partlabel <lx>..	使用这个标签 glob 保存分区。
--save-partindex <id>…	使用这个数值或者范围保存分区。
--offline	强制离线安装。
--insecure	跳过签名验证。
--insecure-ignition	允许没有 HTTPS 或 hash 的 Ignition URL。
--architecture <name>	目标 CPU 架构。默认为 **x86_64**。
--preserve-on-error	出现错误时不清除分区表。
-h，--help	打印帮助信息。
命令行参数	
参数	**描述**
<device>	目的设备。
CoreOS-installer 嵌入的 Ignition 命令	
命令	**描述**

$ coreos-installer iso ignition embed <options> --ignition-file <file_path> <ISO_image>	在 ISO 镜像中嵌入 Ignition 配置。
coreos-installer iso ignition show <options> <ISO_image>	显示来自 ISO 镜像的内嵌 Ignition 配置。
coreos-installer iso ignition remove <options> <ISO_image>	从 ISO 镜像中删除嵌入的 Ignition 配置。

coreos-installer ISO Ignition 选项

选项	描述
-f, --force	覆盖现有的 Ignition 配置。
-i, --ignition-file <path>	要使用的 Ignition 配置。默认为 **stdin**。
-o, --output <path>	将 ISO 写入到一个新输出文件。
-h，--help	打印帮助信息。

coreos-installer PXE Ignition 命令

命令	描述
请注意，不是所有子命令都接受这些选项。	
coreos-installer pxe ignition wrap <options>	在镜像中嵌套 Ignition 配置。
coreos-installer pxe ignition unwrap <options> <image_name>	显示在镜像中嵌套的 Ignition 配置。
coreos-installer pxe ignition unwrap <options> <initrd_name>	在 **initrd** 镜像中显示嵌套的 Ignition 配置。

coreos-installer PXE Ignition 选项

选项	描述
-i, --ignition-file <path>	要使用的 Ignition 配置。默认为 **stdin**。
-o, --output <path>	将 ISO 写入到一个新输出文件。
-h，--help	打印帮助信息。

13.1.12. 创建集群

要创建 OpenShift Container Platform 集群，请等待您通过安装程序生成的 Ignition 配置文件所置备的机器上完成 bootstrap 过程。

先决条件

- 为集群创建所需的基础架构。
- 已获得安装程序并为集群生成了 Ignition 配置文件。
- 已使用 Ignition 配置文件为集群创建 RHCOS 机器。
- 您的机器可直接访问互联网，或者可以使用 HTTP 或 HTTPS 代理。

流程

1. 监控 bootstrap 过程：

```
$ ./openshift-install --dir <installation_directory> wait-for bootstrap-complete \ 1
    --log-level=info 2
```

1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

2 要查看不同的安装详情，请指定 **warn**、**debug** 或 **error**，而不要指定 **info**。

输出示例

```
INFO Waiting up to 30m0s for the Kubernetes API at https://api.test.example.com:6443...
INFO API v1.19.0 up
INFO Waiting up to 30m0s for bootstrapping to complete...
INFO It is now safe to remove the bootstrap resources
```

Kubernetes API 服务器提示已在 control plane 机器上完成 bootstrap 时，命令运行成功。

2. bootstrap 过程完成后，请从负载均衡器中删除 bootstrap 机器。

重要

此时您必须从负载均衡器中删除 bootstrap 机器。您还可以删除或重新格式化机器本身。

13.1.13. 使用 CLI 登录到集群

您可以通过导出集群 **kubeconfig** 文件，以默认系统用户身份登录集群。**kubeconfig** 文件包含关于集群的信息，供 CLI 用于将客户端连接到正确集群和 API 服务器。该文件特只适用于一个特定的集群，在 OpenShift Container Platform 安装过程中创建。

先决条件

- 已部署了 OpenShift Container Platform 集群。
- 已安装 **oc** CLI。

流程

1. 导出 **kubeadmin** 凭证：

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

2. 使用导出的配置，验证能否成功运行 **oc** 命令：

```
$ oc whoami
```

输出示例

```
system:admin
```

13.1.14. 批准机器的证书签名请求

将机器添加到集群时，会为您添加的每台机器生成两个待处理证书签名请求（CSR）。您必须确认这些 CSR 已获得批准，或根据需要自行批准。客户端请求必须首先被批准，然后是服务器请求。

先决条件

- 您已将机器添加到集群中。

流程

1. 确认集群可以识别这些机器：

```
$ oc get nodes
```

输出示例

```
NAME     STATUS   ROLES   AGE  VERSION
master-0 Ready    master  63m  v1.19.0
master-1 Ready    master  63m  v1.19.0
master-2 Ready    master  64m  v1.19.0
```

输出将列出您创建的所有机器。

注意

在一些 CSR 被批准前，以上输出可能不包括计算节点（也称为 worker 节点）。

2. 检查待处理的 CSR，并确保可以看到添加到集群中的每台机器都有 **Pending** 或 **Approved** 状态的客户端请求：

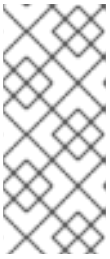
```
$ oc get csr
```

输出示例

```
NAME        AGE    REQUESTOR                                                                   CONDITION
csr-8b2br   15m    system:serviceaccount:openshift-machine-config-operator:node-bootstrapper   Pending
csr-8vnps   15m    system:serviceaccount:openshift-machine-config-operator:node-bootstrapper   Pending
...
```

在本例中，两台机器加入了集群。您可能会在列表中看到更多已批准的 CSR。

3. 如果 CSR 没有获得批准，请在所添加机器的所有待处理 CSR 都处于 **Pending** 状态后，为您的集群机器批准这些 CSR：



注意

由于 CSR 会自动轮转，因此请在将机器添加到集群后一小时内批准您的 CSR。如果没有在一小时内批准，证书将会轮转，每个节点将会存在多个证书。您必须批准所有这些证书。批准客户端 CSR 后， Kubelet 为服务证书创建辅助 CSR，这需要手动批准。然后，如果 Kubelet 请求具有相同参数的新证书，则 **machine-approver** 会自动批准后续服务证书续订请求。

注意

对于在未启用机器 API 的平台中运行的集群，如裸机和其他用户置备的基础架构，必须采用一种方法自动批准 kubelet 提供证书请求（CSR）。如果没有批准请求，则 **oc exec**、**oc rsh** 和 **oc logs** 命令将无法成功，因为 API 服务器连接到 kubelet 时需要服务证书。与 Kubelet 端点联系的任何操作都需要此证书批准。这个方法必须监视新的 CSR，确认 CSR 由 **system:node** 或 **system:admin** 组中的 **node-bootstrapper** 服务帐户提交，并确认节点的身份。

- 若要单独批准，请对每个有效的 CSR 运行以下命令：

```
$ oc adm certificate approve <csr_name> 1
```

1 **<csr_name>** 是当前 CSR 列表中 CSR 的名称。

- 要批准所有待处理的 CSR，请运行以下命令：

```
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}{{"\n"}}{{end}}{{end}}' | xargs --no-run-if-empty oc adm certificate approve
```

注意

在有些 CSR 被批准前，一些 Operator 可能无法使用。

4. 现在，您的客户端请求已被批准，您必须查看添加到集群中的每台机器的服务器请求：

```
$ oc get csr
```

输出示例

```
NAME        AGE    REQUESTOR                                                                   CONDITION
```

```
csr-bfd72   5m26s   system:node:ip-10-0-50-126.us-east-2.compute.internal
Pending
csr-c57lv   5m26s   system:node:ip-10-0-95-157.us-east-2.compute.internal
Pending
...
```

5. 如果剩余的 CSR 没有被批准，且处于 **Pending** 状态，请批准集群机器的 CSR：

   - 若要单独批准，请对每个有效的 CSR 运行以下命令：

     ```
     $ oc adm certificate approve <csr_name> 1
     ```

     1 **<csr_name>** 是当前 CSR 列表中 CSR 的名称。

   - 要批准所有待处理的 CSR，请运行以下命令：

     ```
     $ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}{{"\n"}}
     {{end}}{{end}}' | xargs oc adm certificate approve
     ```

6. 批准所有客户端和服务器 CSR 后，器将处于 **Ready** 状态。运行以下命令验证：

   ```
   $ oc get nodes
   ```

   输出示例

   ```
   NAME      STATUS    ROLES   AGE  VERSION
   master-0  Ready     master  73m  v1.20.0
   master-1  Ready     master  73m  v1.20.0
   master-2  Ready     master  74m  v1.20.0
   worker-0  Ready     worker  11m  v1.20.0
   worker-1  Ready     worker  11m  v1.20.0
   ```

注意

批准服务器 CSR 后可能需要几分钟时间让机器转换为 **Ready** 状态。

其他信息

- 如需有关 CSR 的更多信息，请参阅证书签名请求。

13.1.15. 初始 Operator 配置

在 control plane 初始化后，您必须立即配置一些 Operator 以便它们都可用。

先决条件

- 您的 control plane 已初始化。

流程

1. 观察集群组件上线：

```
$ watch -n5 oc get clusteroperators
```

输出示例

```
NAME                                       VERSION AVAILABLE   PROGRESSING   DEGRADED
SINCE
authentication                             4.6.0   True        False         False      3h56m
cloud-credential                           4.6.0   True        False         False      29h
cluster-autoscaler                         4.6.0   True        False         False      29h
config-operator                            4.6.0   True        False         False      6h39m
console                                    4.6.0   True        False         False      3h59m
csi-snapshot-controller                    4.6.0   True        False         False      4h12m
dns                                        4.6.0   True        False         False      4h15m
etcd                                       4.6.0   True        False         False      29h
image-registry                             4.6.0   True        False         False      3h59m
ingress                                    4.6.0   True        False         False      4h30m
insights                                   4.6.0   True        False         False      29h
kube-apiserver                             4.6.0   True        False         False      29h
kube-controller-manager                    4.6.0   True        False         False      29h
kube-scheduler                             4.6.0   True        False         False      29h
kube-storage-version-migrator              4.6.0   True        False         False      4h2m
machine-api                                4.6.0   True        False         False      29h
machine-approver                           4.6.0   True        False         False      6h34m
machine-config                             4.6.0   True        False         False      3h56m
marketplace                                4.6.0   True        False         False      4h2m
monitoring                                 4.6.0   True        False         False      6h31m
network                                    4.6.0   True        False         False      29h
node-tuning                                4.6.0   True        False         False      4h30m
openshift-apiserver                        4.6.0   True        False         False      3h56m
openshift-controller-manager               4.6.0   True        False         False      4h36m
openshift-samples                          4.6.0   True        False         False      4h30m
operator-lifecycle-manager                 4.6.0   True        False         False      29h
operator-lifecycle-manager-catalog         4.6.0   True        False         False      29h
operator-lifecycle-manager-packageserver   4.6.0   True        False         False      3h59m
service-ca                                 4.6.0   True        False         False      29h
storage                                    4.6.0   True        False         False      4h30m
```

2. 配置不可用的 Operator。

13.1.15.1. 禁用默认的 OperatorHub 源

在 OpenShift Container Platform 安装过程中，默认为 OperatorHub 配置由红帽和社区项目提供的源内容的 operator 目录。在受限网络环境中，必须以集群管理员身份禁用默认目录。

流程

- 通过在 **OperatorHub** 对象中添加 **disableAllDefaultSources: true** 来禁用默认目录的源：

```
$ oc patch OperatorHub cluster --type json \
    -p '[{"op": "add", "path": "/spec/disableAllDefaultSources", "value": true}]'
```

提示

或者，您可以使用 Web 控制台管理目录源。在 **Administration → Cluster Settings → Global Configuration → OperatorHub** 页面中，点 **Sources** 选项卡，其中可创建、删除、禁用和启用单独的源。

13.1.15.2. 安装过程中删除的镜像 registry

在不提供可共享对象存储的平台上，OpenShift Image Registry Operator bootstraps 本身的状态是 **Removed**。这允许 **openshift-installer** 在这些平台类型上完成安装。

将 **ManagementState** Image Registry Operator 配置从 **Removed** 改为 **Managed**。

注意

Prometheus 控制台提供了一个 **ImageRegistryRemoved** 警报，例如：

"Image Registry has been removed.**ImageStreamTags**, **BuildConfigs** and **DeploymentConfigs** which reference **ImageStreamTags** may not work as expected.Please configure storage and update the config to **Managed** state by editing configs.imageregistry.operator.openshift.io."

13.1.15.3. 镜像 registry 存储配置

对于不提供默认存储的平台，Image Registry Operator 最初将不可用。安装后，您必须配置 registry 使用的存储，这样 Registry Operator 才可用。

示配置生产集群所需的持久性卷的说明。如果适用，显示有关将空目录配置为存储位置的说明，该位置只可用于非生产集群。

另外还提供了在升级过程中使用 **Recreate** rollout 策略来允许镜像 registry 使用块存储类型的说明。

13.1.15.3.1. 为裸机和其他手动安装配置 registry 存储

作为集群管理员，在安装后需要配置 registry 来使用存储。

先决条件

- 具有 Cluster Administrator 权限
- 使用手动置备的 Red Hat Enterprise Linux CoreOS（RHCOS）节点（如裸机）的集群。
- 为集群置备的持久性存储，如 Red Hat OpenShift Container Storage。

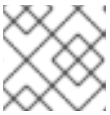
重要

如果您只有一个副本，OpenShift Container Platform 支持对镜像 registry 存储的 **ReadWriteOnce** 访问。要部署支持高可用性的、带有两个或多个副本的镜像 registry，需要 **ReadWriteMany** 访问设置。

- 必须具有 100Gi 容量。

流程

1. 为了配置 registry 使用存储，需要修改 **configs.imageregistry/cluster** 资源中的 **spec.storage.pvc** 。



注意

使用共享存储时，请查看您的安全设置以防止被外部访问。

2. 验证您没有 registry pod:

```
$ oc get pod -n openshift-image-registry
```

注意

如果存储类型为 **emptyDIR**，则副本数不能超过 **1**。

3. 检查 registry 配置：

```
$ oc edit configs.imageregistry.operator.openshift.io
```

输出示例

```
storage:
  pvc:
    claim:
```

将 **claim** 字段留空以允许自动创建一个 **image-registry-storage** PVC。

4. 检查 **clusteroperator** 的状态：

```
$ oc get clusteroperator image-registry
```

5. 确保您的 registry 设置为 manage，以启用镜像的构建和推送。

- 运行：

```
$ oc edit configs.imageregistry/cluster
```

然后将行改

```
managementState: Removed
```

为

```
managementState: Managed
```

13.1.15.3.2. 在非生产集群中配置镜像 registry 存储

您必须为 Image Registry Operator 配置存储。对于非生产集群，您可以将镜像 registry 设置为空目录。如果您这样做，重启 registry 后会丢失所有镜像。

流程

- 将镜像 registry 存储设置为空目录：

```
$ oc patch configs.imageregistry.operator.openshift.io cluster --type merge --patch '{"spec":{"storage":{"emptyDir":{}}}}'
```

警告

仅可为非生产集群配置这个选项。

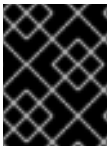
如果在 Image Registry Operator 初始化其组件前运行此命令，**oc patch** 命令会失败并显示以下错误：

```
Error from server (NotFound): configs.imageregistry.operator.openshift.io "cluster" not found
```

等待几分钟，然后再次运行该命令。

13.1.15.3.3. 配置块 registry 存储

要允许镜像 registry 在作为集群管理员升级过程中使用块存储类型，您可以使用 **Recreate** rollout 策略。



重要

支持块存储卷，但不建议将其与生产环境中的镜像 registry 一起使用。在块存储上配置 registry 的安装不具有高可用性，因为 registry 无法拥有多个副本。

流程

1. 要将镜像 registry 存储设置为块存储类型，对 registry 进行补丁，使其使用 **Recreate** rollout 策略，并只使用一个（**1**）副本运行：

```
$ oc patch config.imageregistry.operator.openshift.io/cluster --type=merge -p '{"spec":{"rolloutStrategy":"Recreate","replicas":1}}'
```

2. 为块存储设备置备 PV，并为该卷创建 PVC。请求的块卷使用 ReadWriteOnce（RWO）访问模式。

3. 编辑 registry 配置，使其引用正确的 PVC。

13.1.16. 在用户置备的基础架构上完成安装

完成 Operator 配置后，可以在您提供的基础架构上完成集群安装。

先决条件

- 您的 control plane 已初始化。
- 已完成初始 Operator 配置。

流程

流程

1. 使用以下命令确认所有集群组件都已在线：

```
$ watch -n5 oc get clusteroperators
```

输出示例

```
NAME                                       VERSION AVAILABLE   PROGRESSING   DEGRADED
SINCE
authentication                             4.6.0   True        False         False      3h56m
cloud-credential                           4.6.0   True        False         False      29h
cluster-autoscaler                         4.6.0   True        False         False      29h
config-operator                            4.6.0   True        False         False      6h39m
console                                    4.6.0   True        False         False      3h59m
csi-snapshot-controller                    4.6.0   True        False         False      4h12m
dns                                        4.6.0   True        False         False      4h15m
etcd                                       4.6.0   True        False         False      29h
image-registry                             4.6.0   True        False         False      3h59m
ingress                                    4.6.0   True        False         False      4h30m
insights                                   4.6.0   True        False         False      29h
kube-apiserver                             4.6.0   True        False         False      29h
kube-controller-manager                    4.6.0   True        False         False      29h
kube-scheduler                             4.6.0   True        False         False      29h
kube-storage-version-migrator              4.6.0   True        False         False      4h2m
machine-api                                4.6.0   True        False         False      29h
machine-approver                           4.6.0   True        False         False      6h34m
machine-config                             4.6.0   True        False         False      3h56m
marketplace                                4.6.0   True        False         False      4h2m
monitoring                                 4.6.0   True        False         False      6h31m
network                                    4.6.0   True        False         False      29h
node-tuning                                4.6.0   True        False         False      4h30m
openshift-apiserver                        4.6.0   True        False         False      3h56m
openshift-controller-manager               4.6.0   True        False         False      4h36m
openshift-samples                          4.6.0   True        False         False      4h30m
operator-lifecycle-manager                 4.6.0   True        False         False      29h
operator-lifecycle-manager-catalog         4.6.0   True        False         False      29h
operator-lifecycle-manager-packageserver   4.6.0   True        False         False      3h59m
service-ca                                 4.6.0   True        False         False      29h
storage                                    4.6.0   True        False         False      4h30m
```

或者，通过以下命令，如果所有集群都可用您会接到通知。它还检索并显示凭证：

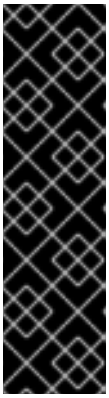
```
$ ./openshift-install --dir <installation_directory> wait-for install-complete 1
```

1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

输出示例

```
INFO Waiting up to 30m0s for the cluster to initialize...
```

Cluster Version Operator 完成从 Kubernetes API 服务器部署 OpenShift Container Platform 集群时，命令运行成功。



重要

- 安装程序生成的 Ignition 配置文件包含在 24 小时后过期的证书，然后在过期时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外情况是，您需要手动批准待处理的 **node-bootstrapper** 证书签名请求（CSR）来恢复 kubelet 证书。如需更多信息，请参阅*从过期的 control plane 证书中恢复*的文档。

- 建议您在生成 12 小时后使用 Ignition 配置文件，因为集群安装后 24 小时证书从 16 小时轮转至 22 小时。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中运行证书更新时避免安装失败。

2. 确认 Kubernetes API 服务器正在与 pod 通信。

   a. 要查看所有 pod 的列表，请使用以下命令：

   ```
   $ oc get pods --all-namespaces
   ```

   输出示例

   ```
   NAMESPACE                       NAME                                         READY   STATUS
   RESTARTS   AGE
   openshift-apiserver-operator    openshift-apiserver-operator-85cb746d55-zqhs8   1/1
   Running   1          9m
   openshift-apiserver             apiserver-67b9g                               1/1     Running   0
   3m
   openshift-apiserver             apiserver-ljcmx                               1/1     Running   0
   1m
   openshift-apiserver             apiserver-z25h4                               1/1     Running   0
   2m
   openshift-authentication-operator authentication-operator-69d5d8bf84-vh2n8     1/1
   Running   0          5m
   ...
   ```

   b. 使用以下命令，查看上一命令的输出中所列 pod 的日志：

   ```
   $ oc logs <pod_name> -n <namespace>
   ```
   (1)

   (1) 指定 pod 名称和命名空间，如上一命令的输出中所示。

   如果 pod 日志显示，Kubernetes API 服务器可以与集群机器通信。

13.1.17. OpenShift Container Platform 的 Telemetry 访问

在 OpenShift Container Platform 4.6 中，默认运行的 Telemetry 服务提供有关集群健康状况和成功更新的指标，需要访问互联网。如果您的集群连接到互联网，Telemetry 会自动运行，而且集群会注册到 OpenShift Cluster Manager。

确认 OpenShift Cluster Manager 清单正确后，可以由 Telemetry 自动维护，也可以使用 OpenShift Cluster Manager 手动维护，使用订阅监控来跟踪帐户或多集群级别的 OpenShift Container Platform 订阅。

其他资源

- 有关 Telemetry 服务的更多信息，请参阅关于远程健康监控。

13.1.18. 后续步骤

- 自定义集群。
- 如果需要，您可以选择不使用远程健康报告。
- 设置 registry 并配置 registry 存储。

第 14 章 安装配置

14.1. 支持的用于不同平台的安装方法

您可以在不同的平台上执行不同类型的安装。

注意

不是所有安装选项都支持所有平台，如下表所示。

表 14.1. 安装程序置备的基础架构选项

	AWS	Azure	GCP	Open Stack	RHV	裸机	vSphere	VMC	IBM Z	IBM Power
默认	X	X	X		X	X	X	X		
Custom	X	X	X	X	X		X	X		
Cluster Network Operator	X	X	X				X	X		
Restricted network	X		X	X			X	X		
私有集群	X	X	X							
现有的虚拟私有网络	X	X	X							
政府区域	X	X								

表 14.2. 用户置备的基础架构

	AWS	Azure	GCP	Open Stack	RHV	裸机	vSphere	VMC	IBM Z	IBM Power
Custom	X	X	X	X	X	X	X	X	X	X

	AWS	Azure	GCP	Open Stack	RHV	裸机	vSphere	VMC	IBM Z	IBM Power
Cluster Network Operator						X	X	X		
Restricted network	X		X			X	X	X	X	X
在集群项目外托管共享 VPC			X							

14.2. 自定义节点

虽然不鼓励直接更改 OpenShift Container Platform 节点，但有时在实现低级别安全、网络或性能功能时需要这样做。通过以下方法可以对 OpenShift Container Platform 节点直接进行更改：

- 创建包含在清单文件中的机器配置，以便在 **openshift-install** 期间启动集群。
- 创建通过 Machine Config Operator 传递至运行的 OpenShift Container Platform 节点的 MachineConfig。

以下小节描述了您可能需要以这种方式在节点中配置的功能。

14.2.1. 添加 day-1 内核参数

虽然修改内核参数通常应做为第 2 天的任务，但您可能希望在初始集群安装过程中将内核参数添加到所有 master 节点或 worker 节点中。下面是一些您可能需要在集群安装过程中添加内核参数以在系统第一次引导前生效的原因：

- 禁用某个功能（比如 SELinux），以使这个功能不会对系统产生任何影响。

警告

不支持在 RHCOS 上禁用 SELinux。

- 您需要在系统启动前进行一些低级网络配置。

要向 master 节点或 worker 节点添加内核参数，您可以创建一个 **MachineConfig** 对象，并将该对象注入 Ignition 在集群设置过程中使用的清单文件集合中。

如需列出引导时可以传递给 RHEL 8 内核的参数，请参阅 Kernel.org 内核参数。如果需要这些参数来完成初始的 OpenShift Container Platform 安装，最好使用这个流程添加内核参数。

流程

1. 切换到包含安装程序的目录，并为集群生成 Kubernetes 清单：

```
$ ./openshift-install create manifests --dir <installation_directory>
```

2. 决定是否要在 worker 或 control plane 节点（也称为 master 节点）中添加内核参数。

3. 在 **openshift** 目录中，创建一个文件（例如: **99-openshift-machineconfig-master-kargs.yaml**）来定义 **MachineConfig** 对象以添加内核设置。这个示例在 control plane 节点中添加了一个 **loglevel=7** 内核参数：

```
$ cat << EOF > 99-openshift-machineconfig-master-kargs.yaml
apiVersion: machineconfiguration.openshift.io/v1
kind: MachineConfig
metadata:
  labels:
    machineconfiguration.openshift.io/role: master
  name: 99-openshift-machineconfig-master-kargs
spec:
  kernelArguments:
    - 'loglevel=7'
EOF
```

   您可以将 **master** 改为 **worker**，以将内核参数添加到 worker 节点。创建一个单独的 YAML 文件，以添加到 master 节点和 worker 节点中。

现在您可以继续创建集群。

14.2.2. 在节点中添加内核模块

对于大多数常用硬件，在计算机启动时，Linux 内核会包括使用这些硬件所需的设备驱动程序模块。但是对于一些硬件来说，在 Linux 中不提供它们的模块。因此，您必须找到一种方法来为每个主机计算机提供这些模块。此流程介绍了如何为 OpenShift Container Platform 集群中的节点进行此操作。

当首先按照这些说明部署了一个内核模块后，这个模块就可用于当前内核。如果安装了新内核，kmods-via-containers 软件将被重建并部署该模块，以便使新内核可使用该模块的兼容版本。

使这个功能可以在每个节点中保持模块最新状态的方法是：

- 在引导时启动的每个节点中添加 systemd 服务，以检测是否安装了新内核。
- 如果检测到一个新的内核，该服务会重建该模块并将其安装到内核中

有关此过程所需软件的详情，请查看 kmods-via-containers github 站点。

需要记住的几个重要问题：

- 这个过程是技术预览。

- 软件工具和示例还没有官方的 RPM，现只能从非官方的 **github.com** 站点获得。
- 红帽不支持您通过这些步骤添加的第三方内核模块。
- 在本流程中，构建您的内核模块所需的软件部署在 RHEL 8 容器中。请记住，当节点有新内核时，每个节点上会自动重新构建模块。因此，每个节点都需要访问一个 **yum** 存储库，该程序存储库包含重建该模块所需的内核和相关软件包。该内容最好由一个有效的 RHEL 订阅提供。

14.2.2.1. 构建并测试内核模块容器

在将内核模块部署到 OpenShift Container Platform 集群之前，您可以在单独的 RHEL 系统中测试此过程。收集内核模块的源代码、KVC 框架和 kmod-via-containers 软件。然后构建并测试模块。要在 RHEL 8 系统中做到这一点，请执行以下操作：

流程

1. 注册 RHEL 8 系统：

```
# subscription-manager register
```

2. 为 RHEL 8 系统添加订阅：

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

3. 安装构建软件和容器所需的软件：

```
# yum install podman make git -y
```

4. 克隆 **kmod-via-containers** 存储库：
   a. 为存储库创建一个文件夹：

```
$ mkdir kmods; cd kmods
```

   b. 克隆存储库：

```
$ git clone https://github.com/kmods-via-containers/kmods-via-containers
```

5. 在 RHEL 8 构建主机上安装 KVC 框架实例来测试模块。这会添加一个 **kmods-via-container** systemd 服务并加载它：
   a. 进入 **kmod-via-containers** 目录：

```
$ cd kmods-via-containers/
```

   b. 安装 KVC 框架实例：

```
$ sudo make install
```

   c. 重新载入 systemd 管理器配置：

```
$ sudo systemctl daemon-reload
```

6. 获取内核模块源代码。通过源代码，可以构建一个您无法控制但由其他人提供的第三方模块。您需要类似 **kvc -simple-kmod** 示例中显示的内容，使用以下步骤将这些内容克隆到您的系统中：

```
$ cd .. ; git clone https://github.com/kmods-via-containers/kvc-simple-kmod
```

7. 编辑示例中的配置文件 **simple-kmod.conf**，将 Dockerfile 的名称改为 **Dockerfile.rhel**：

   a. 进入 **kvc-simple-kmod** 目录：

```
$ cd kvc-simple-kmod
```

   b. 重命名 Dockerfile:

```
$ cat simple-kmod.conf
```

   Dockerfile 示例

```
KMOD_CONTAINER_BUILD_CONTEXT="https://github.com/kmods-via-containers/kvc-simple-kmod.git"
KMOD_CONTAINER_BUILD_FILE=Dockerfile.rhel
KMOD_SOFTWARE_VERSION=dd1a7d4
KMOD_NAMES="simple-kmod simple-procfs-kmod"
```

8. 为您的内核模块创建一个 **kmods-via-containers@.service** 实例（在这个示例中是 **simple-kmod**）：

```
$ sudo make install
```

9. 启用 **kmods-via-containers@.service** 实例：

```
$ sudo kmods-via-containers build simple-kmod $(uname -r)
```

10. 启用并启动 systemd 服务：

```
$ sudo systemctl enable kmods-via-containers@simple-kmod.service --now
```

   a. 查看服务状态：

```
$ sudo systemctl status kmods-via-containers@simple-kmod.service
```

   输出示例

```
● kmods-via-containers@simple-kmod.service - Kmods Via Containers - simple-kmod
Loaded: loaded (/etc/systemd/system/kmods-via-containers@.service;
      enabled; vendor preset: disabled)
Active: active (exited) since Sun 2020-01-12 23:49:49 EST; 5s ago...
```

11. 要确认载入了内核模块，请使用 **lsmod** 命令列出模块：

```
$ lsmod | grep simple_
```

输出示例

```
simple_procfs_kmod     16384  0
simple_kmod            16384  0
```

12. 可选。使用其它方法检查 **simple-kmod** 是否正常工作：

- 使用 **dmesg** 在内核环缓冲中查找 "Hello world" 信息：

```
$ dmesg | grep 'Hello world'
```

输出示例

```
[ 6420.761332] Hello world from simple_kmod.
```

- 在 **/proc** 中检查 **simple-procfs-kmod** 的值：

```
$ sudo cat /proc/simple-procfs-kmod
```

输出示例

```
simple-procfs-kmod number = 0
```

- 运行 **spkut** 命令从模块中获取更多信息：

```
$ sudo spkut 44
```

输出示例

```
KVC: wrapper simple-kmod for 4.18.0-147.3.1.el8_1.x86_64
Running userspace wrapper using the kernel module container...
+ podman run -i --rm --privileged
  simple-kmod-dd1a7d4:4.18.0-147.3.1.el8_1.x86_64 spkut 44
simple-procfs-kmod number = 0
simple-procfs-kmod number = 44
```

下一步，当系统引导这个服务时，会检查新内核是否在运行。如果有一个新内核，该服务会构建内核模块的新版本，然后载入它。如果已经构建了该模块，它将只载入该模块。

14.2.2.2. 为 OpenShift Container Platform 置备内核模块

根据 OpenShift Container Platform 集群首次引导时是否必须存在内核模块，您可以通过以下两种方式之一设置内核模块部署：

- **在集群安装时（day-1）置备内核模块**：您可以通过一个 **MachineConfig** 创建内容，并通过包括一组清单文件来将其提供给 **openshift-install**。
- **通过 Machine Config Operator 置备内核模块（day-2）**：如果可以等到集群启动并运行后再添加内核模块，则可以通过 Machine Config Operator (MCO) 部署内核模块软件。

在这两种情况下，每个节点都需要在检测到新内核时可以获得内核软件包及相关软件包。您可以通过以下几种方法设置每个节点来获取该内容。

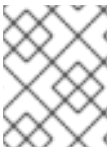
- 为每个节点提供 RHEL 权利。

- 从现有的 RHEL 主机获取 RHEL 权利。把 **/etc/pki/entitlement** 目录中的 RHEL 授权复制到与您在构建 Ignition 配置时提供的其他文件相同的位置。
- 在 Dockerfile 中，添加包括了内核和其他软件包的 **yum** 存储库。这必须包括新内核软件包，因为它们需要与新安装的内核相匹配。

14.2.2.2.1. 通过 MachineConfig 对象置备内核模块

通过将内核模块软件与 **MachineConfig** 对象一起打包，您可以在安装时或通过 Machine Config Operator 向 worker 或 master 节点发送该软件。

首先创建您要使用的基本 Ignition 配置。在安装时，Ignition 配置需要包含添加到集群中的 **core** 用户的 **authorized_keys** 文件中的 ssh 公钥 。如果在以后通过 MCO 添加 **MachineConfig**，则不需要 SSH 公钥。对于这两种方式，simple-kmod 服务示例都会创建一个 systemd 单元文件，它需要一个 **kmods-via-containers@simple-kmod.service**



注意

systemd 单元是 上游程序错误的一个临时解决方案。确保 **kmods-via-containers@simple-kmod.service** 在引导时启动：

1. 注册 RHEL 8 系统：

```
# subscription-manager register
```

2. 为 RHEL 8 系统添加订阅：

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

3. 安装构建软件所需的软件：

```
# yum install podman make git -y
```

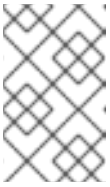
4. 创建 Ignition 配置文件以创建 systemd 单元文件：
   a. 创建用于托管 Ignition 配置文件的目录：

```
$ mkdir kmods; cd kmods
```

   b. 创建 Ignition 配置文件以创建 systemd 单元文件：

```
$ cat <<EOF > ./baseconfig.ign
{
  "ignition": { "version": "3.1.0" },
  "passwd": {
    "users": [
      {
        "name": "core",
        "groups": ["sudo"],
        "sshAuthorizedKeys": [
          "ssh-rsa AAAA"
        ]
      }
    ]
```

```
  },
  "systemd": {
    "units": [{
      "name": "require-kvc-simple-kmod.service",
      "enabled": true,
      "contents": "[Unit]\nRequires=kmods-via-containers@simple-
kmod.service\n[Service]\nType=oneshot\nExecStart=/usr/bin/true\n\n[Install]\nWantedBy=m
ulti-user.target"
    }]
  }
}
EOF
```



注意

您必须将公共 SSH 密钥添加到 **baseconfig.ign** 文件中，以便 **openshift-install** 使用该文件。如果使用 MCO 创建 **MachineConfig** 对象，则不需要公共 SSH 密钥。

5. 创建使用以下配置的基本 MCO YAML 片断：

```
$ cat <<EOF > mc-base.yaml
apiVersion: machineconfiguration.openshift.io/v1
kind: MachineConfig
metadata:
  labels:
    machineconfiguration.openshift.io/role: worker
  name: 10-kvc-simple-kmod
spec:
  config:
EOF
```

注意

Mc-base.yaml 设置为在 **worker** 节点上部署内核模块。要部署到 master 节点上，请将角色从 **worker** 更改为 **master**。要进行这两个操作，您可以通过为不同类型的部署使用不同的文件名来重复整个过程。

6. 获得 **kmods-via-containers** 软件：

   a. 克隆 **kmod-via-containers** 存储库：

   ```
   $ git clone https://github.com/kmods-via-containers/kmods-via-containers
   ```

   b. 克隆 **kvc-simple-kmod** 存储库：

   ```
   $ git clone https://github.com/kmods-via-containers/kvc-simple-kmod
   ```

7. 获取您的模块软件。在这个示例中使用 **kvc-simple-kmod**：

8. 使用之前克隆的存储库，创建一个 fakeroot 目录，并将您要通过 Ignition 传递的文件放置到这个目录：

   a. 创建目录：

```
$ FAKEROOT=$(mktemp -d)
```

b. 进入 **kmod-via-containers** 目录：

```
$ cd kmods-via-containers
```

c. 安装 KVC 框架实例：

```
$ make install DESTDIR=${FAKEROOT}/usr/local CONFDIR=${FAKEROOT}/etc/
```

d. 进入 **kvc-simple-kmod** 目录：

```
$ cd ../kvc-simple-kmod
```

e. 创建实例：

```
$ make install DESTDIR=${FAKEROOT}/usr/local CONFDIR=${FAKEROOT}/etc/
```

9. 获取名为 **filetranspiler** 的工具程序及相关的依赖软件：

```
$ cd .. ; sudo yum install -y python3
git clone https://github.com/ashcrow/filetranspiler.git
```

10. 生成最终机器配置 YAML（**mc.yaml**），其中包括基本 Ignition 配置、基础机器配置以及包括您要提供的文件的 fakeroot 目录：

```
$ ./filetranspiler/filetranspile -i ./baseconfig.ign \
    -f ${FAKEROOT} --format=yaml --dereference-symlinks \
    | sed 's/^/     /' | (cat mc-base.yaml -) > 99-simple-kmod.yaml
```

11. 如果集群还没有启用，生成清单文件并将该文件添加到 **openshift** 目录中。如果集群已在运行，按照以下方法应用该文件：

```
$ oc create -f 99-simple-kmod.yaml
```

您的节点将启动 **kmods-via-containers@simple-kmod.service** 服务，并将载入内核模块。

12. 为了确认内核模块已加载，您可以登录到节点（使用 **oc debug node/<openshift-node>**，然后运行 **chroot /host**）。要列出模块，请使用 **lsmod** 命令：

```
$ lsmod | grep simple_
```

输出示例

```
simple_procfs_kmod     16384  0
simple_kmod            16384  0
```

14.2.3. 在安装过程中加密磁盘

您可以在安装时在 control plane 和计算节点上为引导磁盘启用加密。OpenShift Container Platform 支持 Trusted Platform 模块 (TPM) v2 和 Tang 加密模式。

- TPM v2：这是首选模式。TPM v2 将密码短语存储在服务器中包含的安全加密处理器中。如果从服务器中删除了磁盘，您可以使用这个模式防止集群节点中的引导磁盘数据被解密。
- Tang：Tang 和 Clevis 是启用网络绑定磁盘加密 (NBDE) 的服务器和客户端组件。您可以将集群节点中的引导磁盘数据绑定到 Tang 服务器。这可以防止数据被解密，除非节点位于可访问 Tang 服务器的安全网络中。Clevis 是一种自动化的解密框架，用于在客户端实现解密。

重要

使用 Tang 加密模式加密您的磁盘只支持在用户置备的基础架构上的裸机和 vSphere 安装。

启用 TPM v2 或 Tang 加密模式时，RHCOS 引导磁盘将使用 LUKS2 格式进行加密。

这个功能：

- 可用于安装程序置备的基础架构和用户置备的基础架构部署
- 只在 Red Hat Enterprise Linux CoreOS (RHCOS) 系统上被支持
- 在清单安装阶段设置磁盘加密，以便加密所有写入磁盘的数据（从第一次引导开始）
- 只加密 root 文件系统中的数据（/**dev/mapper/coreos-luks-root** on /）
- 不需要用户干预就可以提供密码短语
- 使用 AES-256-CBC 加密

按照以下两个流程之一为集群中的节点启用磁盘加密。

14.2.3.1. 启用 TPM v2 磁盘加密

使用以下步骤在 OpenShift Container Platform 安装过程中启用 TPM v2 模式磁盘加密。

先决条件

- 您已在安装节点上下载了 OpenShift Container Platform 安装程序。

流程

1. 检查每个节点的 BIOS 是否需要启用 TPM v2 加密。这在大多数 Dell 系统中是必需的。请参阅您的计算机的相关手册。
2. 在安装节点上，切换到包含安装程序的目录，并为集群生成 Kubernetes 清单：

```
$ ./openshift-install create manifests --dir <installation_directory> 1
```

1 将 **<installation_directory>** 替换为您要存储安装文件的目录的路径。

3. 使用 TPM v2 加密模式为 control plane 或计算节点创建机器配置文件来加密引导磁盘。

- 要在 control plane 节点上配置加密，请将以下机器配置示例保存到 **<installation_directory>/openshift** 目录中的一个文件中。例如，将文件命名为 **99-openshift-master-tpmv2-encryption.yaml** ：

```
apiVersion: machineconfiguration.openshift.io/v1
kind: MachineConfig
metadata:
  name: master-tpm
  labels:
    machineconfiguration.openshift.io/role: master
spec:
  config:
    ignition:
      version: 3.1.0
    storage:
      files:
      - contents:
          source: data:text/plain;base64,e30K
        mode: 420
        overwrite: true
        path: /etc/clevis.json
```

- 要在计算节点上配置加密，请将以下机器配置示例保存到 **<installation_directory>/openshift** 目录中的文件中。例如，将文件命名为 **99-openshift-worker-tpmv2-encryption.yaml**：

```
apiVersion: machineconfiguration.openshift.io/v1
kind: MachineConfig
metadata:
  name: worker-tpm
  labels:
    machineconfiguration.openshift.io/role: worker
spec:
  config:
    ignition:
      version: 3.1.0
    storage:
      files:
      - contents:
          source: data:text/plain;base64,e30K
        mode: 420
        overwrite: true
        path: /etc/clevis.json
```

4. 创建 YAML 文件的备份副本。创建 Ignition 配置文件时会消耗原始 YAML 文件。

5. 继续进行 OpenShift Container Platform 安装的其余部分。

14.2.3.2. 启用 Tang 磁盘加密

使用以下步骤在 OpenShift Container Platform 安装过程中启用 Tang 模式磁盘加密。

先决条件

- 您已在安装节点上下载了 OpenShift Container Platform 安装程序。

- 您可以使用 Red Hat Enterprise Linux (RHEL) 8 机器来生成 Tang Exchange 密钥的指纹。

流程

1. 设置 Tang 服务器或访问现有服务器。具体步骤请查看网络绑定磁盘加密 。
2. 添加内核参数来配置集群在安装 Red Hat Enterprise Linux CoreOS（RHCOS）时的网络。例如：在内核命令行中添加参数来配置 DHCP 网络，指定 **ip=dhcp**，或者设置静态网络。对于 DHCP 和静态网络，您必须提供 **rd.neednet=1** 内核参数。

重要

跳过这一步会导致第二次引导失败。

4. 如果尚未安装，在 RHEL 8 机器上安装 **clevis** 软件包：

```
$ sudo yum install clevis
```

5. 在 RHEL 8 计算机上，运行以下命令来生成交换密钥的指纹。使用 Tang 服务器的 URL 替换 **http://tang.example.com:7500**：

```
$ clevis-encrypt-tang '{"url":"http://tang.example.com:7500"}' < /dev/null > /dev/null 1
```

1 在本例中，**tangd.socket** 侦听 Tang 服务器上的端口 **7500**。

注意

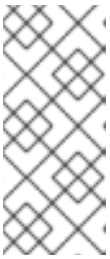
此步骤中仅使用 **clevis-encrypt-tang** 命令，以生成交换密钥的指纹。此时不会将数据传递给命令以进行加密，因此 **/dev/null** 作为输入而不是纯文本提供。加密的输出也会发送到 **/dev/null**，因为此过程不需要它。

输出示例

```
The advertisement contains the following signing keys:

PLjNyRdGw03zlRoGjQYMahSZGu9 1
```

1 Exchange 键的指纹。

当出现 **Do you wish to trust these keys? [ynYN]** 提示时，输入 **Y**。



注意

RHEL 8 提供 Clevis 版本 15，它使用 SHA-1 哈希算法来生成指纹。些其他发行版提供 Clevis 版本 17 或更高版本，它们使用 SHA-256 哈希算法进行指纹。您必须使用 SHA-1 创建 thumbprint 的 Clevis 版本来防止在 OpenShift Container Platform 集群节点上安装 Red Hat Enterprise Linux CoreOS (RHCOS) 时出现 Clevis 绑定问题。

6. 创建 Base64 编码文件，使用新生成的 Tang server 替换 **url**，thumbprint 替换**thp**：

```
-
```

```
$ (cat <<EOM
{
 "url": "http://tang.example.com:7500", (1)
 "thp": "PLjNyRdGw03zlRoGjQYMahSZGu9" (2)
}
EOM
) | base64 -w0
```

(1) 指定 Tang 服务器的 URL。在本例中，**tangd.socket** 侦听 Tang 服务器上的端口 **7500**。

(2) 指定上一步中生成的 Exchange key thumbprint。

输出示例

```
ewogInVybCI6ICJodHRwOi8vdGFuZy5leGFtcGxlLmNvbTo3NTAwIiwgCiAidGhwIjogIlBMak55
UmRHdzAzemxSb0dqUVlNYWhTWkd1OSIgCn0K
```

7. 如果您还没有生成 Kubernetes 清单，请切换到安装节点上包含安装程序的目录，并创建它们：

输出示例

```
$ ./openshift-install create manifests --dir <installation_directory> (1)
```

(1) 将 **<installation_directory>** 替换为您要存储安装文件的目录的路径。

8. 使用 Tang 加密模式，为 control plane 或计算节点创建机器配置文件来加密引导磁盘。

- 要在 control plane 节点上配置加密，请将以下机器配置示例保存到 **<installation_directory>/openshift** 目录中的一个文件中。例如，将文件命名为 **99-openshift-master-tang-encryption.yaml**：

```
apiVersion: machineconfiguration.openshift.io/v1
kind: MachineConfig
metadata:
  name: master-tang
  labels:
    machineconfiguration.openshift.io/role: master
spec:
  config:
    ignition:
      version: 3.1.0
    storage:
      files:
      - contents:
          source: data:text/plain;base64,e30K
          source:
data:text/plain;base64,ewogInVybCI6ICJodHRwOi8vdGFuZy5leGFtcGxlLmNvbTo3NTAwIi
wgCiAidGhwIjogIlBMak55UmRHdzAzemxSb0dqUVlNYWhTWkd1OSIgCn0K (1)
        mode: 420
        overwrite: true
```

```
        path: /etc/clevis.json
    kernelArguments:
      - rd.neednet=1 2
```

1 指定在上一步中生成的以 Base64 编码的字符串。

2 添加 **rd.neednet=1** 内核参数，以在 initramfs 中启动网络。此参数是必需的。

- 要在计算节点上配置加密，请将以下机器配置示例保存到 **<installation_directory>/openshift** 目录中的文件中。例如，将文件命名为 **99-openshift-worker-tang-encryption.yaml**：

```
apiVersion: machineconfiguration.openshift.io/v1
kind: MachineConfig
metadata:
  name: worker-tang
  labels:
    machineconfiguration.openshift.io/role: worker
spec:
  config:
    ignition:
      version: 3.1.0
    storage:
      files:
      - contents:
          source: data:text/plain;base64,e30K
          source:
data:text/plain;base64,ewogInVybCI6ICJodHRwOi8vdGFuZy5leGFtcGxlLmNvbTo3NTAwIi
wgCiAidGhwIjogIlBMak55UmRHdzAzemxSb0dqUVlNYWhTWkd1OSIgCn0K 1
        mode: 420
        overwrite: true
        path: /etc/clevis.json
  kernelArguments:
    - rd.neednet=1 2
```

1 指定在上一步中生成的以 Base64 编码的字符串。

2 添加 **rd.neednet=1** 内核参数，以在 initramfs 中启动网络。此参数是必需的。

9. 创建 YAML 文件的备份副本。创建 Ignition 配置文件时会消耗原始 YAML 文件。
10. 继续进行 OpenShift Container Platform 安装的其余部分。

14.2.4. 配置启用了 RAID 的数据卷

您可以启用软件 RAID 分区以提供外部数据卷。

流程

- 在 < **installation_directory>/openshift** 目录中创建机器配置，以使用软件 RAID 配置数据卷。在本例中，该文件名为 **raid1-alt-storage.yaml**：

  辅助磁盘配置文件上的 RAID 1 示例

```
apiVersion: machineconfiguration.openshift.io/v1
kind: MachineConfig
metadata:
  labels:
    machineconfiguration.openshift.io/role: worker
  name: raid1-alt-storage
spec:
  config:
    ignition:
      version: 3.1.0
    storage:
      disks:
      - device: /dev/sdc
        partitions:
        - label: data-1
        wipeTable: true
      - device: /dev/sdd
        partitions:
        - label: data-2
        wipeTable: true
      filesystems:
      - device: /dev/md/md-data
        format: xfs
        path: /var/lib/containers
        wipeFilesystem: true
      raid:
      - devices:
        - /dev/disk/by-partlabel/data-1
        - /dev/disk/by-partlabel/data-2
        level: raid1
        name: md-data
    systemd:
      units:
      - contents: |-
          [Unit]
          Before=local-fs.target
          Requires=systemd-fsck@dev-md-md\x2ddata.service
          After=systemd-fsck@dev-md-md\x2ddata.service

          [Mount]
          Where=/var/lib/containers
          What=/dev/md/md-data
          Type=xfs

          [Install]
          RequiredBy=local-fs.target
        enabled: true
        name: var-lib-containers.mount (1)
```

(1) 如果选择不同的挂载点，您必须更新单元名称以匹配您的挂载点。否则，该单元将不激活。您可以使用 **echo $(systemd-escape -p $mountpoint)生成匹配的单元名称。mount**，其中 **$mountpoint** 是您选择的挂载点。

14.2.5. 配置 chrony 时间服务

您可以通过修改 **chrony.conf** 文件的内容来设置 chrony 时间服务（**chronyd**）使用的时间服务器和相关设置，并通过一个机器配置将这些内容传递给节点。

流程

1. 创建 **chrony.conf** 文件的内容并对其进行 base64 编码。例如：

```
$ cat << EOF | base64
   pool 0.rhel.pool.ntp.org iburst 1
   driftfile /var/lib/chrony/drift
   makestep 1.0 3
   rtcsync
   logdir /var/log/chrony
EOF
```

1 指定任何有效的、可访问的时间源，如 DHCP 服务器提供的时间源。另外，您可以指定以下 NTP 服务器：**1.rhel.pool.ntp.org**、**2.rhel.pool.ntp.org** 或 **3.rhel.pool.ntp.org**。

输出示例

```
ICAgIHNlcnZlciBjbG9jay5yZWRoYXQuY29tIGlidXJzdAogICAgZHJpZnRmaWxlIC92YXIvbGli
L2Nocm9ueS9kcmlmdAogICAgbWFrZXN0ZXAgMS4wIDMKICAgIHJ0Y3N5bmMKICAgIGxvZ2Rpci
RpciAv
dmFyL2xvZy9jaHJvbnkK
```

2. 创建 **MachineConfig** 对象文件，将 base64 字符串替换为您刚刚创建的字符串。本例将文件添加到 **master** 节点。您可以将其更改为 **worker**，或为 **worker** 角色创建额外的 MachineConfig。为集群使用的每种机器创建 MachineConfig 文件：

```
$ cat << EOF > ./99-masters-chrony-configuration.yaml
apiVersion: machineconfiguration.openshift.io/v1
kind: MachineConfig
metadata:
  labels:
    machineconfiguration.openshift.io/role: master
  name: 99-masters-chrony-configuration
spec:
  config:
    ignition:
      config: {}
      security:
        tls: {}
      timeouts: {}
      version: 3.1.0
    networkd: {}
    passwd: {}
    storage:
      files:
      - contents:
          source: data:text/plain;charset=utf-
8;base64,ICAgIHNlcnZlciBjbG9jay5yZWRoYXQuY29tIGlidXJzdAogICAgZHJpZnRmaWxlIC92Y
XIvbGliL2Nocm9ueS9kcmlmdAogICAgbWFrZXN0ZXAgMS4wIDMKICAgIHJ0Y3N5bmMKICAg
IGxvZ2RpciAvdmFyL2xvZy9jaHJvbnkK
        mode: 420 1
```

```
      overwrite: true
      path: /etc/chrony.conf
  osImageURL: ""
EOF
```

1 为机器配置文件的 **mode** 字段指定数值模式。在创建文件并应用更改后，**模式** 将转换为十进制值。您可以使用 **oc get mc <mc-name> -o yaml** 命令来检查 YAML 文件。

3. 对配置文件做一个备份副本。
4. 使用两种方式之一应用配置：
   - 如果集群还没有启动，在生成清单文件后，将此文件添加到 **<installation_directory>/openshift** 目录中，然后继续创建集群。
   - 如果集群已在运行，请应用该文件：

```
$ oc apply -f ./99-masters-chrony-configuration.yaml
```

14.2.6. 其他资源

- 如需更多与 FIPS 相关的信息，请参阅 Support for FIPS cryptography。

14.3. 可用的集群自定义

大多数集群配置和自定义在 OpenShift Container Platform 集群部署后完成。有若干*配置资源*可用。

您可以修改配置资源来配置集群的主要功能，如镜像 registry、网络配置、镜像构建操作以及用户身份供应商。

如需设置这些资源的当前信息，请使用 **oc explain** 命令，如 **oc explain builds --api-version=config.openshift.io/v1**

14.3.1. 集群配置资源

所有集群配置资源都作用于全局范围（而非命名空间），且命名为 **cluster**。

资源名称	描述
apiserver.config.openshift.io	提供 API 服务器配置，如证书和证书颁发机构。
authentication.config.openshift.io	控制集群的身份提供程序和身份验证配置。
build.config.openshift.io	控制集群中所有构建的默认和强制配置。
console.config.openshift.io	配置 Web 控制台界面的行为，包括注销行为。

资源名称	描述
featuregate.config.openshift.io	启用 FeatureGates，以便您能使用技术预览功能。
image.config.openshift.io	配置应如何对待特定的镜像 registry（允许、禁用、不安全、CA 详情）。
ingress.config.openshift.io	与路由相关的配置详情，如路由的默认域。
oauth.config.openshift.io	配置用户身份供应商，以及与内部 OAuth 服务器流程相关的其他行为。
project.config.openshift.io	配置项目的创建方式，包括项目模板。
proxy.config.openshift.io	定义需要外部网络访问的组件要使用的代理。注意：目前不是所有组件都会消耗这个值。
scheduler.config.openshift.io	配置调度程序行为，如策略和默认节点选择器。

14.3.2. Operator 配置资源

这些配置资源是集群范围的实例，即 **cluster**，控制归特定 Operator 所有的特定组件的行为。

资源名称	描述
consoles.operator.openshift.io	控制控制台外观，如品牌定制
config.imageregistry.operator.openshift.io	配置内部镜像 registry 设置，如公共路由、日志级别、代理设置、资源约束、副本数和存储类型。
config.samples.operator.openshift.io	配置 Samples Operator，以控制在集群上安装哪些镜像流和模板示例。

14.3.3. 其他配置资源

这些配置资源代表一个特定组件的单一实例。在有些情况下，您可以通过创建多个资源实例来请求多个实例。在其他情况下，Operator 只消耗指定命名空间中的特定资源实例名称。如需有关如何和何时创建其他资源实例的详情，请参考具体组件的文档。

资源名称	实例名称	命名空间	描述
alertmanager.monitoring.coreos.com	**main**	**openshift-monitoring**	控制 Alertmanager 部署参数。
ingresscontroller.operator.openshift.io	**default**	**openshift-ingress-operator**	配置 Ingress Operator 行为，如域、副本数、证书和控制器放置。

14.3.4. 信息资源

可以使用这些资源检索集群信息。请不要直接编辑这些资源。

资源名称	实例名称	描述
clusterversion.config.openshift.io	**version**	在 OpenShift Container Platform 4.6 中，不得自定义生产集群的 **ClusterVersion** 资源。相反，请按照 更新集群的流程 进行操作。
dns.config.openshift.io	**cluster**	无法修改集群的 DNS 设置。您可以查看 DNS Operator 状态。
infrastructure.config.openshift.io	**cluster**	允许集群与其云供应商交互的配置详情。
network.config.openshift.io	**cluster**	无法在安装后修改集群网络。要自定义您的网络，请遵循相关的流程在安装过程中自定义网络。

14.3.5. 更新全局集群 pull secret

您可以通过替换当前的 pull secret 或附加新的 pull secret 来更新集群的全局 pull secret。

当用户使用单独的 registry 存储镜像时，需要这个过程，而不是在安装过程中使用的 registry。

警告

集群资源必须调整为新的 pull secret，这样可暂时限制集群的可用性。

警告

更新全局 pull secret 会导致在 Machine Config Operator（MCO）同步更改时节点重启。

先决条件

- 您可以使用具有 **cluster-admin** 角色的用户访问集群。

流程

1. 可选：要将新的 pull secret 附加到现有 pull secret 中，请完成以下步骤：
   a. 输入以下命令下载 pull secret：

```
$ oc get secret/pull-secret -n openshift-config --template='{{index .data
".dockerconfigjson" | base64decode}}' ><pull_secret_location> 1
```

   1 提供 pull secret 文件的路径。

   b. 输入以下命令来添加新 pull secret：

```
$ oc registry login --registry="<registry>" \ 1
--auth-basic="<username>:<password>" \ 2
--to=<pull_secret_location> 3
```

   1 提供新的 registry。您可以在同一个 registry 中包含多个软件仓库，例如：**--registry="<registry/my-namespace/my-repository>"**。

   2 提供新 registry 的凭据。

   3 提供 pull secret 文件的路径。

   另外，您可以对 pull secret 文件执行手动更新。

2. 输入以下命令为您的集群更新全局 pull secret：

```
$ oc set data secret/pull-secret -n openshift-config --from-file=.dockerconfigjson=
<pull_secret_location> 1
```

   1 提供新 pull secret 文件的路径。

该更新将推广至所有节点，可能需要一些时间，具体取决于集群大小。在这段时间中，节点会排空（drain），pod 将在剩余节点上重新调度。

14.4. 配置防火墙

如果使用防火墙，您必须进行配置，以便 OpenShift Container Platform 能访问正常运作所需要的网站。您必须始终授予一些站点的访问权限，如果使用 Red Hat Insights、Telemetry 服务、托管集群的云以及某些构建策略，则还要授予更多站点的访问权限。

14.4.1. 为 OpenShift Container Platform 配置防火墙

在安装 OpenShift Container Platform 之前，您必须配置防火墙，以授予 OpenShift Container Platform 所需站点的访问权限。

和在 worker 节点上运行的服务相比，运行在控制器节点中的服务没有特殊的配置注意事项。

流程

1. 允许以下 registry URL：

URL	端口	功能
registry.redhat.io	443, 80	提供核心容器镜像
quay.io	443, 80	提供核心容器镜像
***.quay.io**	443, 80	提供核心容器镜像
***.openshiftapps.com**	443, 80	提供 Red Hat Enterprise Linux CoreOS（RHCOS）镜像

   当您将 **quay.io** 等网站添加到 allowlist 中时，不要在您的 denylist 中添加通配符条目，如 ***.quay.io**。在大多数情况下，镜像 registry 使用内容交付网络（CDN）来提供镜像。如果防火墙阻止访问，则当初始下载请求重定向到一个主机名（如 **cdn01.quay.io** ）时，镜像下载将被拒绝。

   当您在 allowlist 中添加通配符条目时（如 ***.quay.io**），会涵盖 CDN 主机名（如 **cdn01.quay.io**）。

2. 将提供构建所需语言或框架的资源的任何站点列入允许列表。

3. 如果不禁用 Telemetry，您必须授予对以下 URL 的访问权限，以便能访问 Red Hat Insights：

URL	端口	功能
cert-api.access.redhat.com	443, 80	Telemetry 所需
api.access.redhat.com	443, 80	Telemetry 所需
infogw.api.openshift.com	443, 80	Telemetry 所需
console.redhat.com/api/ingress	443, 80	Telemetry 和 **insights-operator** 需要

4. 如果使用 Amazon Web Services (AWS)、Microsoft Azure 或 Google Cloud Platform (GCP) 来托管您的集群，您必须授予对提供该云的云供应商 API 和 DNS 的 URL 的访问权限：

云	URL	端口	功能
AWS	***.amazonaws.com**	443, 80	需要此项以访问 AWS 服务和资源。请参阅 AWS 文档中 AWS Service Endpoints，以确定您使用的区域所允许的具体端点。
GCP	***.googleapis.com**	443, 80	需要此项以访问 GCP 服务和资源。请参阅 GCP 文档中的 Cloud Endpoints，以确定您的 API 所允许的端点。
	accounts.google.com	443, 80	需要此项以访问您的 GCP 帐户。
Azure	**management.azure.com**	443, 80	需要此项以访问 Azure 服务和资源。请参阅 Azure 文档中的 Azure REST API 参考，以确定您的 API 所允许的端点。
	***.blob.core.windows.net**	443, 80	需要此项以下载 Ignition 文件。
	login.microsoftonline.com	443, 80	需要此项以访问 Azure 服务和资源。请参阅 Azure 文档中的 Azure REST API 参考，以确定您的 API 所允许的端点。

5. 将以下 URL 列入允许列表：

URL	端口	功能
mirror.openshift.com	443, 80	需要此项以访问镜像安装内容和镜像。此站点也是发行镜像签名的来源，但 Cluster Version Operator 只需要一个可正常工作的源。
storage.googleapis.com/openshift-release	443, 80	发行版本镜像签名源，但 Cluster Version Operator 只需要一个可正常工作的源。
***.apps.<cluster_name>.<base_domain>**	443, 80	需要此项以访问默认的集群路由，除非您在安装过程中设置了入口通配符。
quayio-production-s3.s3.amazonaws.com	443, 80	需要此项以访问 AWS 中的 Quay 镜像内容。
api.openshift.com	443, 80	集群令牌需要，并检查集群是否有可用的更新。
art-rhcos-ci.s3.amazonaws.com	443, 80	需要此项以下载 Red Hat Enterprise Linux CoreOS（RHCOS）镜像。

URL	端口	功能
console.redhat.com/openshift	443, 80	集群令牌所需
registry.access.redhat.com	443, 80	**odo** CLI 需要。
sso.redhat.com	443, 80	**https://console.redhat.com/openshift** 站点使用来自 **sso.redhat.com** 的身份验证

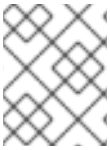
操作员需要路由访问权限来执行健康检查。特别是，身份验证和 Web 控制台 Operator 会连接到两个路由，以验证路由是否正常工作。如果您是集群管理员，且不想允许 ***.apps.<cluster_name>.<base_domain>**，请允许这些路由：

- **oauth-openshift.apps.<cluster_name>.<base_domain>**
- **console-openshift-console.apps.<cluster_name>.<base_domain>**，或者，在 **consoles.operator/cluster** 对象的 **spec.route.hostname** 字段中指定的主机名（如果此字段非空）。

6. 将以下 URL 列入允许的可选第三方内容：

URL	端口	功能
registry.connect.redhat.com	443, 80	所有第三方镜像和认证操作器必填.
rhc4tp-prod-z8cxf-image-registry-us-east-1-evenkyleffocxqvofrk.s3.dualstack.us-east-1.amazonaws.com	443, 80	提供对托管在 **registry.connect.redhat.com**上的容器镜像的访问
oso-rhc4tp-docker-registry.s3-us-west-2.amazonaws.com	443, 80	对于 Sonatype Nexus、F5 大 IP 操作器是必需的。

7. 如果您使用默认的 Red Hat Network Time Protocol（NTP）服务器允许以下 URL：
   - **1.rhel.pool.ntp.org**
   - **2.rhel.pool.ntp.org**
   - **3.rhel.pool.ntp.org**



注意

如果您没有使用默认的 Red Hat NTP 服务器，请验证您的平台的 NTP 服务器并在您的防火墙中允许它。

14.5. 配置私有集群

安装 OpenShift Container Platform 版本 4.6 集群后，您可以将其某些核心组件设置为私有。

14.5.1. 关于私有集群

默认情况下，OpenShift Container Platform 被置备为使用可公开访问的 DNS 和端点。在部署集群后，您可以将 DNS、Ingress Controller 和 API 服务器设置为私有。

DNS
如果在安装程序置备的基础架构上安装 OpenShift Container Platform，安装程序会在预先存在的公共区中创建记录，并在可能的情况下为集群自己的 DNS 解析创建一个私有区。在公共区和私有区中，安装程序或集群为 ***.apps** 和 **Ingress** 对象创建 DNS 条目，并为 API 服务器创建 **api**。

公共和私有区中的 ***.apps** 记录是相同的，因此当您删除公有区时，私有区为集群无缝地提供所有 DNS 解析。

Ingress Controller
由于默认 **Ingress** 对象是作为公共对象创建的，所以负载均衡器是面向互联网的，因此在公共子网中。您可以将默认 Ingress Controller 替换为内部控制器。

API Server
默认情况下，安装程序为 API 服务器创建适当的网络负载均衡器，供内部和外部流量使用。

在 Amazon Web Services（AWS）上，会分别创建独立的公共和私有负载均衡器。负载均衡器是基本相同的，唯一不同是带有一个额外的、用于在集群内部使用的端口。虽然安装程序根据 API 服务器要求自动创建或销毁负载均衡器，但集群并不管理或维护它们。只要保留集群对 API 服务器的访问，您可以手动修改或移动负载均衡器。对于公共负载均衡器，需要打开端口 6443，并根据 **/readyz** 路径配置 HTTPS 用于健康检查。

在 Google Cloud Platform 上，会创建一个负载均衡器来管理内部和外部 API 流量，因此您无需修改负载均衡器。

在 Microsoft Azure 上，会创建公共和私有负载均衡器。但是，由于当前实施的限制，您刚刚在私有集群中保留两个负载均衡器。

14.5.2. 将 DNS 设置为私有

部署集群后，您可以修改其 DNS 使其只使用私有区。

流程

1. 查看集群的 **DNS** 自定义资源：

```
$ oc get dnses.config.openshift.io/cluster -o yaml
```

输出示例

```
apiVersion: config.openshift.io/v1
kind: DNS
metadata:
  creationTimestamp: "2019-10-25T18:27:09Z"
  generation: 2
  name: cluster
  resourceVersion: "37966"
```

```
  selfLink: /apis/config.openshift.io/v1/dnses/cluster
  uid: 0e714746-f755-11f9-9cb1-02ff55d8f976
spec:
  baseDomain: <base_domain>
  privateZone:
    tags:
      Name: <infrastructure_id>-int
      kubernetes.io/cluster/<infrastructure_id>: owned
  publicZone:
    id: Z2XXXXXXXXXXA4
status: {}
```

请注意，**spec** 部分包含一个私有区和一个公共区。

2. 修补 **DNS** 自定义资源以删除公共区：

```
$ oc patch dnses.config.openshift.io/cluster --type=merge --patch='{"spec": {"publicZone":
null}}'
dns.config.openshift.io/cluster patched
```

因为 Ingress Controller 在创建 **Ingress** 对象时会参考 **DNS** 定义，因此当您创建或修改 **Ingress** 对象时，只会创建私有记录。

重要

在删除公共区时，现有 Ingress 对象的 DNS 记录不会修改。

3. 可选：查看集群的 **DNS** 自定义资源，并确认已删除公共区：

```
$ oc get dnses.config.openshift.io/cluster -o yaml
```

输出示例

```
apiVersion: config.openshift.io/v1
kind: DNS
metadata:
  creationTimestamp: "2019-10-25T18:27:09Z"
  generation: 2
  name: cluster
  resourceVersion: "37966"
  selfLink: /apis/config.openshift.io/v1/dnses/cluster
  uid: 0e714746-f755-11f9-9cb1-02ff55d8f976
spec:
  baseDomain: <base_domain>
  privateZone:
    tags:
      Name: <infrastructure_id>-int
      kubernetes.io/cluster/<infrastructure_id>-wfpg4: owned
status: {}
```

14.5.3. 将 Ingress Controller 设置为私有

部署集群后，您可以修改其 Ingress Controller 使其只使用私有区。

流程

1. 修改默认 Ingress Controller，使其仅使用内部端点：

```
$ oc replace --force --wait --filename - <<EOF
apiVersion: operator.openshift.io/v1
kind: IngressController
metadata:
  namespace: openshift-ingress-operator
  name: default
spec:
  endpointPublishingStrategy:
    type: LoadBalancerService
    loadBalancer:
      scope: Internal
EOF
```

输出示例

```
ingresscontroller.operator.openshift.io "default" deleted
ingresscontroller.operator.openshift.io/default replaced
```

删除公共 DNS 条目，并更新私有区条目。

14.5.4. 将 API 服务器限制为私有

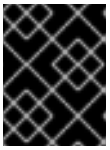
将集群部署到 Amazon Web Services（AWS）或 Microsoft Azure 后，可以重新配置 API 服务器，使其只使用私有区。

先决条件

- 安装 OpenShift CLI（**oc**）。
- 使用具有 **admin** 权限的用户登陆到 web 控制台。

流程

1. 在 AWS 或 Azure 的 web 门户或控制台中，执行以下操作：
   a. 找到并删除相关的负载均衡器组件。
      - 对于 AWS，删除外部负载均衡器。私有区的 API DNS 条目已指向内部负载均衡器，它使用相同的配置，因此您无需修改内部负载均衡器。
      - 对于 Azure，删除负载均衡器的 **api-internal** 规则。
   b. 在公共区中删除 **api.$clustername.$yourdomain** DNS 条目。
2. 删除外部负载均衡器：



重要

您只能对安装程序置备的基础架构 (IPI) 集群执行以下步骤。对于用户置备的基础架构 (UPI) 集群，您必须手动删除或禁用外部负载均衡器。

a. 在终端中列出集群机器：

```
$ oc get machine -n openshift-machine-api
```

输出示例

```
NAME                                 STATE     TYPE       REGION      ZONE         AGE
lk4pj-master-0                       running   m4.xlarge  us-east-1   us-east-1a   17m
lk4pj-master-1                       running   m4.xlarge  us-east-1   us-east-1b   17m
lk4pj-master-2                       running   m4.xlarge  us-east-1   us-east-1a   17m
lk4pj-worker-us-east-1a-5fzfj        running   m4.xlarge  us-east-1   us-east-1a   15m
lk4pj-worker-us-east-1a-vbghs        running   m4.xlarge  us-east-1   us-east-1a   15m
lk4pj-worker-us-east-1b-zgpzg        running   m4.xlarge  us-east-1   us-east-1b   15m
```

在以下步骤中，修改 control plane 机器（名称中包含 **master** 的机器）。

b. 从每台 control plane 机器移除外部负载均衡器。

i. 编辑 control plane **Machine** 对象以移除对外部负载均衡器的引用：

```
$ oc edit machines -n openshift-machine-api <master_name> 1
```

1 指定要修改的 control plane 或 master **Machine** 对象的名称。

ii. 删除描述外部负载均衡器的行（在以下示例中已被标记），保存并退出对象规格：

```
...
spec:
  providerSpec:
    value:
      ...
        loadBalancers:
        - name: lk4pj-ext 1
          type: network 2
        - name: lk4pj-int
          type: network
```

1 2 删除这一行。

iii. 对名称中包含 **master** 的每个机器重复这个过程。

第 15 章 验证安装

您可以按照本文档中的步骤在安装后检查 OpenShift Container Platform 集群的状态。

15.1. 查看安装日志

您可以在 OpenShift Container Platform 安装日志中查看安装概述。如果安装成功，日志中包含访问集群所需的信息。

先决条件

- 有访问安装主机的访问权限。

流程

- 查看安装主机上安装目录中的 **.openshift_install.log** 日志文件：

```
$ cat <install_dir>/.openshift_install.log
```

输出示例

如果安装成功，则日志末尾包含集群凭证，如下例所示：

```
...
time="2020-12-03T09:50:47Z" level=info msg="Install complete!"
time="2020-12-03T09:50:47Z" level=info msg="To access the cluster as the system:admin
user when using 'oc', run 'export KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'"
time="2020-12-03T09:50:47Z" level=info msg="Access the OpenShift web-console here:
https://console-openshift-console.apps.mycluster.example.com"
time="2020-12-03T09:50:47Z" level=info msg="Login to the console with user: \"kubeadmin\",
and password: \"6zYIx-ckbW3-4d2Ne-IWvDF\""
time="2020-12-03T09:50:47Z" level=debug msg="Time elapsed per stage:"
time="2020-12-03T09:50:47Z" level=debug msg="    Infrastructure: 6m45s"
time="2020-12-03T09:50:47Z" level=debug msg="Bootstrap Complete: 11m30s"
time="2020-12-03T09:50:47Z" level=debug msg=" Bootstrap Destroy: 1m5s"
time="2020-12-03T09:50:47Z" level=debug msg=" Cluster Operators: 17m31s"
time="2020-12-03T09:50:47Z" level=info msg="Time elapsed: 37m26s"
```

15.2. 查看镜像拉取源

对于没有网络连接的集群，您可以使用节点上的命令来查看拉取的镜像源，如 **crictl images**。

但是，对于断开连接的安装，若要查看拉取镜像的来源，您必须查看 CRI-O 日志以查找 **Trying to access** 日志条目，如下所示。查看镜像拉取源的其他方法，如 **crictl images** 命令，显示非镜像镜像的镜像名称，即使镜像是从镜像位置拉取的。

先决条件

- 您可以使用具有 **cluster-admin** 角色的用户访问集群。

流程

- 查看 master 或 worker 节点的 CRI-O 日志：

```
$  oc adm node-logs <node_name> -u crio
```

输出示例

Trying to access 日志条目指示镜像要从中拉取的位置。

```
...
Mar 17 02:52:50 ip-10-0-138-140.ec2.internal crio[1366]: time="2021-08-05
10:33:21.594930907Z" level=info msg="Pulling image: quay.io/openshift-release-dev/ocp-
release:4.8.4-ppc64le" id=abcd713b-d0e1-4844-ac1c-474c5b60c07c
name=/runtime.v1alpha2.ImageService/PullImage
Mar 17 02:52:50 ip-10-0-138-140.ec2.internal crio[1484]: time="2021-03-17
02:52:50.194341109Z" level=info msg="Trying to access \"li0317gcp1.mirror-
registry.qe.gcp.devcluster.openshift.com:5000/ocp/release@sha256:1926eae7cacb9c00f142ec
98b00628970e974284b6ddaf9a6a086cb9af7a6c31\""
Mar 17 02:52:50 ip-10-0-138-140.ec2.internal crio[1484]: time="2021-03-17
02:52:50.226788351Z" level=info msg="Trying to access \"li0317gcp1.mirror-
registry.qe.gcp.devcluster.openshift.com:5000/ocp/release@sha256:1926eae7cacb9c00f142ec
98b00628970e974284b6ddaf9a6a086cb9af7a6c31\""
...
```

日志可能会显示镜像拉取源两次，如上例中所示。

如果您的 **ImageContentSourcePolicy** 对象列出了多个镜像，OpenShift Container Platform 会尝试按照配置中列出的顺序拉取镜像，例如：

```
Trying to access \"li0317gcp1.mirror-
registry.qe.gcp.devcluster.openshift.com:5000/ocp/release@sha256:1926eae7cacb9c00f142ec
98b00628970e974284b6ddaf9a6a086cb9af7a6c31\"
Trying to access \"li0317gcp2.mirror-
registry.qe.gcp.devcluster.openshift.com:5000/ocp/release@sha256:1926eae7cacb9c00f142ec
98b00628970e974284b6ddaf9a6a086cb9af7a6c31\"
```

15.3. 获取集群版本、状态和更新详情

您可以通过运行 **oc get clusterversion** 命令来查看集群版本和状态。如果状态显示安装仍在进行，您可以查看 Operator 的状态以了解更多信息。

您还可以列出当前的更新频道并查看可用的集群更新。

先决条件

- 您可以使用具有 **cluster-admin** 角色的用户访问集群。
- 已安装 OpenShift CLI（**oc**）。

流程

1. 获取集群版本和整体状态：

```
$ oc get clusterversion
```

输出示例

-

```
NAME     VERSION   AVAILABLE   PROGRESSING   SINCE   STATUS
version  4.6.4     True        False         6m25s   Cluster version is 4.6.4
```

示例输出显示集群已被成功安装。

2. 如果集群状态表示安装仍在进行，您可以通过检查 Operator 状态来获取更详细的进度信息：

```
$ oc get clusteroperators.config.openshift.io
```

3. 查看集群规格、更新可用性和更新历史记录的详细概述：

```
$ oc describe clusterversion
```

4. 列出当前的更新频道：

```
$ oc get clusterversion -o jsonpath='{.items[0].spec}{"\n"}'
```

输出示例

```
{"channel":"stable-4.6","clusterID":"245539c1-72a3-41aa-9cec-
72ed8cf25c5c","upstream":"https://api.openshift.com/api/upgrades_info/v1/graph"}
```

5. 查看可用的集群更新：

```
$ oc adm upgrade
```

输出示例

```
Cluster version is 4.6.4

Updates:

VERSION IMAGE
4.6.6   quay.io/openshift-release-dev/ocp-
release@sha256:c7e8f18e8116356701bd23ae3a23fb9892dd5ea66c8300662ef30563d7104f3
9
```

其他资源

- 如需了解有关查询 Operator 状态的更多信息，请参阅在安装后查询 Operator 状态。
- 如需有关调查 Operator 问题的信息，请参阅故障排除 Operator 的问题。
- 有关更新集群的更多信息，请参阅更新集群。
- 如需有关升级 发行频道的概述，请参阅 OpenShift Container Platform 升级 频道和发行版本。

15.4. 使用 CLI 查询集群节点状态

您可以在安装后验证集群节点的状态。

先决条件

- 您可以使用具有 **cluster-admin** 角色的用户访问集群。
- 已安装 OpenShift CLI（**oc**）。

流程

1. 列出集群节点的状态。验证输出是否列出所有预期的 control plane 和计算节点，以及每个节点的状态是否为 **Ready**：

```
$ oc get nodes
```

输出示例

```
NAME                         STATUS   ROLES    AGE   VERSION
compute-1.example.com        Ready    worker   33m   v1.19.0+9f84db3
control-plane-1.example.com  Ready    master   41m   v1.19.0+9f84db3
control-plane-2.example.com  Ready    master   45m   v1.19.0+9f84db3
compute-2.example.com        Ready    worker   38m   v1.19.0+9f84db3
compute-3.example.com        Ready    worker   33m   v1.19.0+9f84db3
control-plane-3.example.com  Ready    master   41m   v1.19.0+9f84db3
```

2. 查看每个集群节点的 CPU 和内存资源的可用性：

```
$ oc adm top nodes
```

输出示例

```
NAME                         CPU(cores)   CPU%   MEMORY(bytes)   MEMORY%
compute-1.example.com        128m         8%     1132Mi          16%
control-plane-1.example.com  801m         22%    3471Mi          23%
control-plane-2.example.com  1718m        49%    6085Mi          40%
compute-2.example.com        935m         62%    5178Mi          75%
compute-3.example.com        111m         7%     1131Mi          16%
control-plane-3.example.com  942m         26%    4100Mi          27%
```

其他资源

- 如需了解更多有关检查节点健康状况和调查节点问题的详细信息，请参阅验证节点健康状况。

15.5. 从 OPENSHIFT CONTAINER PLATFORM WEB 控制台查看集群状态

您可以参阅 OpenShift Container Platform Web 控制台中的 **Overview** 页面中的以下信息：

- 集群的一般状态
- control plane、集群 Operator 和存储的状态
- CPU、内存、文件系统、网络传输和 pod 可用性
- 集群的 API 地址、集群 ID 和供应商名称
- 集群版本信息

- 集群更新状态，包括当前更新频道和可用更新的详情
- 详细节点、pod、存储类和持久性卷声明（PVC）信息的集群清单
- 持续的集群活动和最近事件列表

先决条件

- 您可以使用具有 **cluster-admin** 角色的用户访问集群。

流程

- 在 **Administrator** 视角中，导航到 **Home → Overview**。

15.6. 查看 RED HAT OPENSHIFT CLUSTER MANAGER 中的集群状态

您可以在 OpenShift Cluster Manager 中查看有关集群状态的详细信息。

先决条件

- 您可以使用具有 **cluster-admin** 角色的用户访问集群。

流程

1. 在 **Administrator** 视角中，导航到 **Home → Overview → Details → OpenShift Cluster Manager**，打开 OpenShift Cluster Manager 中的集群的 **Overview** 页面。

注意

或者，您可以直接导航到 OpenShift Cluster Manager，再从可用集群列表中选择集群 ID。

2. 在 **Overview** 页面中，查看有关集群的以下信息：
   - vCPU 和内存可用性和资源使用情况
   - 集群 ID、状态、类型、位置和供应商名称
   - 按节点类型划分的节点数
   - 集群版本详情、集群的创建日期和集群所有者的名称
   - 集群的生命周期支持状态
   - 订阅信息，包括服务等级协议（SLA）状态、订阅单元类型、集群的生产环境状态、订阅保障和服务等级
   - 集群历史记录
3. 导航到 **Monitoring** 页面查看以下信息：
   - 已检测到的问题列表
   - 正在触发的警报列表

- 集群 Operator 状态和版本
- 集群资源使用量

4. 导航到 **Insights** 页面，查看 Red Hat Insights 提供的以下信息：
   - 集群可能会暴露的问题，按风险程度分类
   - 根据分类的健康检查状态

其他资源

- 如需了解更多与检查集群中的潜在问题的信息，请参阅使用 Insights 发现集群中的问题。

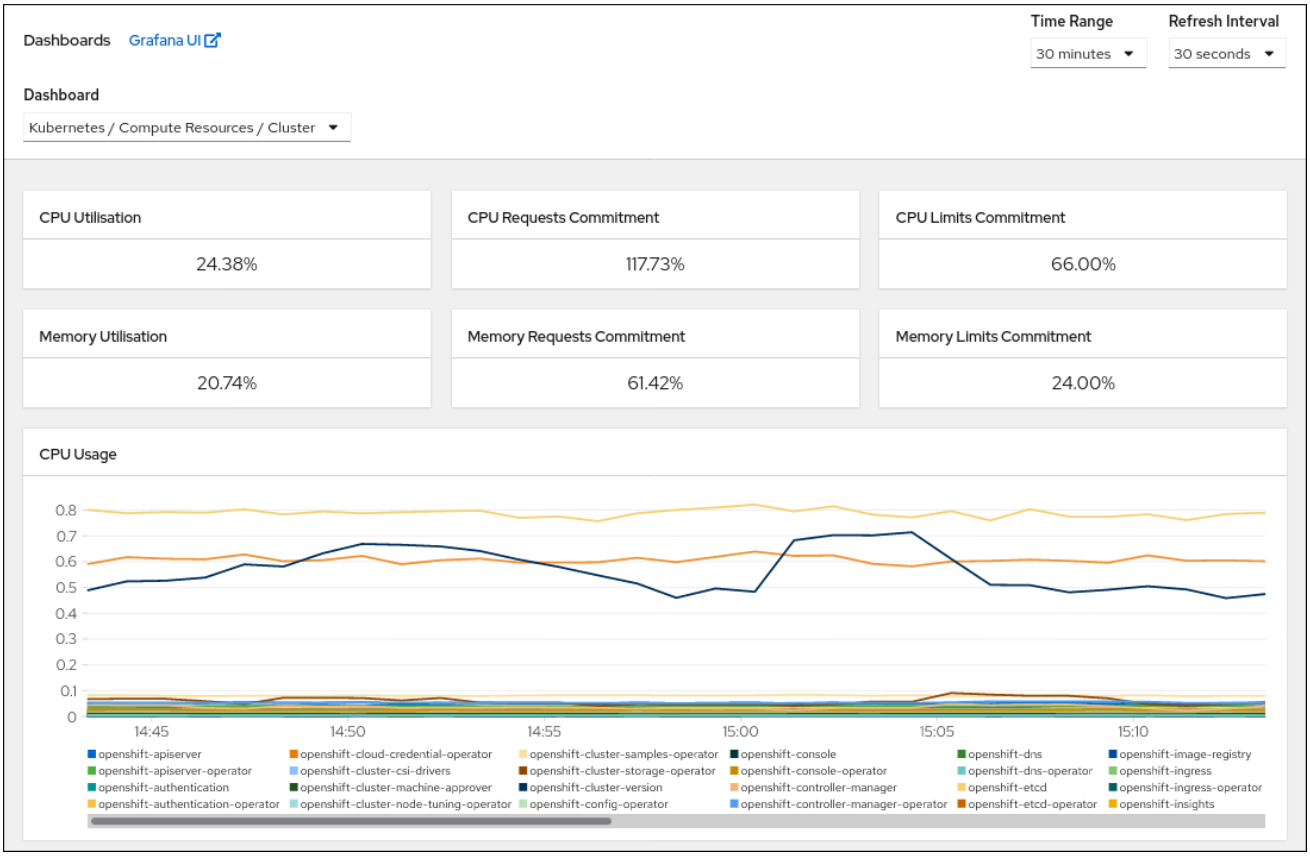
15.7. 检查集群资源可用性和使用

OpenShift Container Platform 提供了一组完整的监控仪表板，帮助您了解集群组件的状态。

在 **Administrator** 视角中，您可以访问 OpenShift Container Platform 核心组件的仪表板，包括：

- etcd
- Kubernetes 计算资源
- Kubernetes 网络资源
- Prometheus
- 与集群和节点性能相关的仪表板

图 15.1. 计算资源仪表板示例



先决条件

- 您可以使用具有 **cluster-admin** 角色的用户访问集群。

流程

1. 在 OpenShift Container Platform Web 控制台内的 **Administrator** 视角中，导航到 **Operators → Dashboards**。
2. 在 **Dashboard** 列表中选择一个仪表板。有些仪表板（如 **etcd** 仪表板）在被选择时会生成额外的子菜单。
3. 可选：在 **Time Range** 列表中为图形选择一个时间范围。
4. 可选：选择一个 **Refresh Interval**。
5. 将鼠标悬停在仪表板中的每个图形上，以显示具体项目的详细信息。

其他资源

- 如需有关 OpenShift Container Platform 监控堆栈的更多信息，请参阅监控概述。

15.8. 列出正在触发的警报

当 OpenShift Container Platform 集群中有一组定义的条件满足时，警报会提供通知。您可以使用 OpenShift Container Platform Web 控制台中的 Alerting UI 查看集群中触发的警报。

先决条件

- 您可以使用具有 **cluster-admin** 角色的用户访问集群。

流程

1. 在 **Administrator** 视角中，导航到 **Monitoring → Alerting → Alerts** 页面。
2. 查看正在触发的警报，包括 **严重性**、**状态** 和**源**。
3. 在 **Alert Details** 页面中选择一个警报来查看更详细的信息。

其他资源

- 如需了解更多与 OpenShift Container Platform 中警报相关的信息，请参阅管理警报。

15.9. 后续步骤

- 如果您在安装集群时遇到问题，请参阅对安装进行障排除。
- 安装 OpenShift Container Platform 后，您可以进一步扩展和自定义集群。

第 16 章 安装问题的故障排除

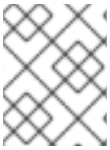
为帮助排查 OpenShift Container Platform 安装失败的问题，您可以从 bootstrap 和 control plane 或 master 机器中收集日志。您还可以从安装程序获得调试信息。

16.1. 先决条件

- 已尝试安装 OpenShift Container Platform 集群，但安装失败。

16.2. 从失败安装中收集日志

如果为安装程序提供了 SSH 密钥，则可以收集与失败安装相关的数据。



注意

用于收集失败安装日志的命令与从正在运行的集群收集日志时所用的命令不同。如果需要从正在运行的集群收集日志，请使用 **oc adm must-gather** 命令。

先决条件

- OpenShift Container Platform 安装在 bootstrap 过程完成前失败。bootstrap 节点正在运行，并可通过 SSH 访问。
- **ssh-agent** 进程在您的计算机上处于活跃状态，并且为 **ssh-agent** 进程和安装程序提供了相同的 SSH 密钥。
- 如果尝试在您置备的基础架构上安装集群，则必须具有 bootstrap 和 control plane 节点（也称为 master 节点）的完全限定域名。

流程

1. 生成从 bootstrap 和 control plane 机器获取安装日志的命令：
   - 如果您使用安装程序置备的基础架构，请切换到包含安装程序的目录，并运行以下命令：

     ```
     $ ./openshift-install gather bootstrap --dir <installation_directory> 1
     ```

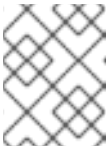
     1 **installation_directory** 是您在运行 **./openshift-install create cluster** 时指定的目录。这个目录包括了安装程序所创建的 OpenShift Container Platform 定义文件。

     对于安装程序置备的基础架构，安装程序会保存有关集群的信息，因此您不用指定主机名或 IP 地址。

   - 如果您使用您置备的基础架构，切换到包含安装程序的目录，并运行以下命令：

     ```
     $ ./openshift-install gather bootstrap --dir <installation_directory> \ 1
       --bootstrap <bootstrap_address> \ 2
       --master <master_1_address> \ 3
       --master <master_2_address> \ 4
       --master <master_3_address>" 5
     ```

1. 对于 **installation_directory**，请指定在运行 **./openshift-install create cluster** 时指定的同一目录。这个目录包括了安装程序所创建的 OpenShift Container Platform 定义文件。
2. **<bootstrap_address>** 是集群 bootstrap 机器的完全限定域名或 IP 地址。
3. 4. 5. **<master_address>** 是集群中 control plane 或 master 机器的完全限定域名或 IP 地址。



注意

默认集群包含三个 control plane 机器。如所示，列出所有 control plane 机器，无论集群使用了多少个。

输出示例

```
INFO Pulling debug logs from the bootstrap machine
INFO Bootstrap gather logs captured here "<installation_directory>/log-bundle-
<timestamp>.tar.gz"
```

如果需要创建关安装失败的红帽支持问题单，请在问题单中附上压缩日志。

16.3. 通过到主机的 SSH 连接手动收集日志

在 **must-gather** 或自动收集方法无法正常工作的情况下手动收集日志。

先决条件

- 必须有到主机的 SSH 访问权限。

流程

1. 运行以下命令，使用 **journalctl** 命令从 bootstrap 主机收集 **bootkube.service** 服务日志：

```
$ journalctl -b -f -u bootkube.service
```

2. 使用 podman logs 命令收集 bootstrap 主机的容器日志。以下命令从主机获取所有容器的日志：

```
$ for pod in $(sudo podman ps -a -q); do sudo podman logs $pod; done
```

3. 或者，通过运行以下命令来使用 **tail** 命令收集主机的容器日志：

```
# tail -f /var/lib/containers/storage/overlay-containers/*/userdata/ctr.log
```

4. 运行 **journalctl** 命令从 master 和 worker 主机收集 **kubelet.service** 和 **crio.service** 服务日志：

```
$ journalctl -b -f -u kubelet.service -u crio.service
```

5. 使用 **tail** 命令收集 master 和 worker 主机容器日志：

```
$ sudo tail -f /var/log/containers/*
```

16.4. 在不使用 SSH 连接到主机的情况下手动收集日志

在 **must-gather** 或自动收集方法无法正常工作的情况下手动收集日志。

如果您无法对节点进行 SSH 访问，则可以通过访问系统日志来调查主机上发生的情况。

先决条件

- OpenShift Container Platform 安装已完成。
- API 服务仍然可以正常工作。
- 有系统管理员特权。

流程

1. 通过运行以下命令访问 **/var/log** 中的 **journald** 单元日志：

```
$ oc adm node-logs --role=master -u kubelet
```

2. 通过运行以下命令访问 **/var/log** 中的主机文件路径：

```
$ oc adm node-logs --role=master --path=openshift-apiserver
```

16.5. 从安装程序获取调试信息

您可以使用以下方法从安装程序获取调试信息。

- 在隐藏的 **.openshift_install.log** 文件中查看与过去安装相关的 debug 信息。例如，输入：

```
$ cat ~/<installation_directory>/.openshift_install.log 1
```

1 对于 **installation_directory**，请指定在运行 **./openshift-install create cluster** 时指定的同一目录。

- 进入包含安装程序的目录，并使用 **--log-level=debug** 重新运行它：

```
$ ./openshift-install create cluster --dir <installation_directory> --log-level debug 1
```

1 对于 **installation_directory**，请指定在运行 **./openshift-install create cluster** 时指定的同一目录。

16.6. 重新安装 OPENSHIFT CONTAINER PLATFORM 集群

如果您无法调试并解决失败的 OpenShift Container Platform 安装中的问题，请考虑安装新的 OpenShift Container Platform 集群。在再次开始安装过程前，您必须完成彻底的清理。对于用户置备的基础架构(UPI)安装，您必须手动销毁集群并删除所有关联的资源。以下流程用于安装程序置备的基础架构(IPI)安装。

流程

1. 销毁集群并删除与集群关联的所有资源，包括安装目录中的隐藏安装程序状态文件：

```
$ ./openshift-install destroy cluster --dir <installation_directory> 1
```

1 **installation_directory** 是您在运行 **./openshift-install create cluster** 时指定的目录。这个目录包括了安装程序所创建的 OpenShift Container Platform 定义文件。

2. 在重新安装集群前，删除安装目录：

```
$ rm -rf <installation_directory>
```

3. 按照安装新 OpenShift Container Platform 集群的步骤进行操作。

第 17 章 支持 FIPS 加密

您可以在 **x86_64** 架构中使用 FIPS 验证的/Modules in Process 加密库安装 OpenShift Container Platform 集群。

对于集群中的 Red Hat Enterprise Linux CoreOS (RHCOS) 机器，当机器根据 **install-config.yaml** 文件中的选项的状态进行部署时，会应用这个更改，该文件管理用户可在集群部署过程中更改的集群选项。在 Red Hat Enterprise Linux（RHEL）机器中，您必须在计划用作 worker 的机器上安装操作系统时，启用 FIPS 模式。这些配置方法可确保集群满足 FIPS 合规审核的要求：在初始系统引导前，只启用经 FIPS 验证/Modules in Process 加密的软件包。

因为 FIPS 必须在集群首次引导操作系统之前启用，所以您不能在部署集群后启用 FIPS。

17.1. OPENSHIFT CONTAINER PLATFORM 中的 FIPS 验证

OpenShift Container Platform 在 Red Hat Enterprise Linux (RHEL) 和 Red Hat CoreOS (RHCOS) 中使用特定的 FIPS 验证/Modules in Process 模块用于使用它们的操作系统组件。请参阅 RHEL7 core crypto components 中的内容。例如，当用户 SSH 到 OpenShift Container Platform 集群和容器时，这些连接会被正确加密。

OpenShift Container Platform 组件以 Go 编写，并使用红帽的 golang 编译器构建。当您为集群启用 FIPS 模式时，所有需要加密签名的 OpenShift Container Platform 组件调用 RHEL 和 RHCOS 加密程序库。

表 17.1. OpenShift Container Platform 4.6 中的 FIPS 模式属性和限制

属性	限制：
RHEL 7 操作系统支持 FIPS 。	FIPS 实现还没有提供一个单一的计算哈希函数和验证基于该哈希的键的函数。在以后的 OpenShift Container Platform 版本中，将继续评估并改进这个限制。
CRI-O 运行时支持 FIPS。	
OpenShift Container Platform 服务支持 FIPS。	
FIPS 验证的/Modules in Process 的加密模块和算法，它们从 RHEL 7 和 RHCOS 二进制文件和镜像中获得。	
使用 FIPS 兼容 golang 编译器。	对 TLS FIPS 的支持当前还不完整，但计划在将来的 OpenShift Container Platform 版本中被完全支持。
在多个构架间支持 FIPS。	目前，仅在使用 **x86_64** 架构的 OpenShift Container Platform 部署中支持 FIPS。

17.2. 集群使用的组件支持 FIPS

尽管 OpenShift Container Platform 集群本身使用 FIPS 验证的/Modules in Process 模块，但请确保支持 OpenShift Container Platform 集群的系统也使用 FIPS 验证的/Modules in Process 模块进行加密。

17.2.1. etcd

要确保存储在 etcd 中的 secret 在进程加密中使用 FIPS 验证的/Modules in Process 模块，请以 FIPS 模式引导节点。在使用 FIPS 模式安装集群后，您可以使用 FIPS 批准的 **aes cbc** 加密算法加密 etcd 数据。

17.2.2. Storage

对于本地存储，使用 RHEL 提供的磁盘加密或者使用 RHEL 提供的磁盘加密的容器原生存储。通过将所有数据存储到使用 RHEL 提供的磁盘加密的卷中，并为您的集群启用 FIPS 模式，静态数据和正在启动的数据或网络数据都受到 FIPS 验证的/Modules in Process 加密的保护。您可以将集群配置为加密每个节点的根文件系统，如自定义节点 中所述。

17.2.3. 运行时

要确保容器知道它们在使用 FIPS 验证的/Modules in Process 加密模块的主机上运行，请使用 CRI-O 管理您的运行时。CRI-O 支持 FIPS 模式，它将容器配置为知道它们是在 FIPS 模式下运行的。

17.3. 在 FIPS 模式下安装集群

要使用 FIPS 模式安装集群，请按照在相应的基础架构中安装自定义集群的步骤进行。在部署集群前，请确定在 **install-config.yaml** 文件中设置了 **fips: true**。

- Amazon Web Services
- Microsoft Azure
- 裸机
- Google Cloud Platform
- Red Hat OpenStack Platform (RHOSP)
- VMware vSphere

注意

如果使用 Azure File 存储，则无法启用 FIPS 模式。

要将 **AES CBC** 加密应用到 etcd 数据存储中，请在安装集群后执行加密 etcd 数据操作。

如果您在集群中添加 RHEL 节点，请确定在机器初始引导前启用 FIPS 模式。请参阅 RHEL 7 文档中的 Adding RHEL compute machines to a OpenShift Container Platform cluster 和 Enabling FIPS Mode 部分。